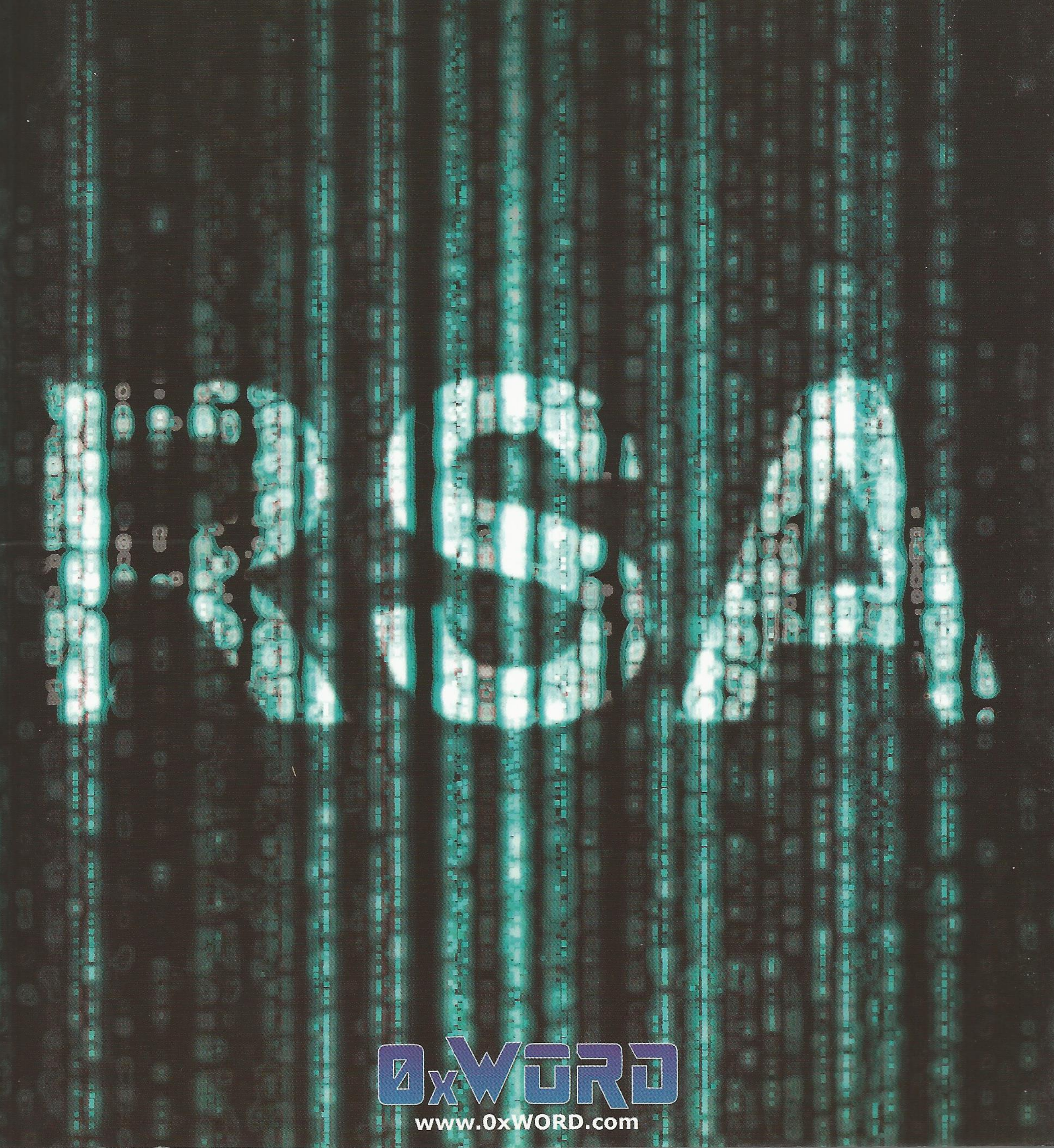
Cifrado de las comunicaciones digitales
De la cifra clásica al algoritmo RSA

Alfonso Muñoz Muñoz y Jorge Ramió Aguirre





Cifrado de las comunicaciones digitales. De la cifra clásica al algoritmo RSA

Alfonso Muñoz y Jorge Ramió

Índice

Prólogo

Bienvenido a mi mundo, el mundo de lo enigmático y misterioso. El mundo de los secretos velados, el mundo en el cual las matemáticas sirven para apoyar movimientos civiles, sincronizar cazas para bombardear aldeas perdidas o establecer comunicaciones privadas entre miembros del Vaticano de las cuales ni su mismo Dios querría saber. El libro que está a punto de leer profundiza en el apasionante mundo de la criptología.

Tal vez la primera pregunta que se haya hecho al leer el título de este libro "Cifrado de las comunicaciones digitales. De la cifra clásica al algoritmo RSA" es qué tienen en común aquellos ingeniosos artilugios de cifra de clave única (simétricos) de antes de la mitad del siglo XX con el estándar actual de la cifra con clave pública o asimétrica RSA en pleno siglo XXI. Parecen a simple vista dos mundos muy lejanos pero la verdad es que ambos se nos presentan como verdaderos paradigmas de la criptografía, al menos si nos atenemos a su trascendencia en la protección de la información, cada uno en su época claro está, y a la cantidad de años que los primeros resistieron al criptoanálisis y que, en su caso, lo viene haciendo RSA.

Muchos de los sistemas de cifra clásica cumplían con creces el principio de caducidad relacionado con la criptografía, en tanto resultaba difícil en aquellos tiempos su criptoanálisis -aunque hoy nos parezca algo trivial- y en cuanto al criptosistema RSA, creado en 1978 por *Rivest*, *Shamir* y *Adleman*, éste se publica como estándar PKCS #1 en 1991. Ha resistido por tanto más de 34 años como algoritmo público y otros 21 años como estándar mundial, y nada hace presagiar que le queden pocos años de vida dado su amplio uso en protocolos de comunicaciones seguras. Todo un récord que supera al otro algoritmo más longevo de la criptografía moderna *Data Encryption Standard* DES, declarado estándar de cifra simétrica en 1976 y que sucumbe definitivamente en 1999, es decir a los 23 años.

Es por ello que este libro se dedica especialmente a estos dos paradigmas de la criptografía, la clásica y RSA, y los trata a fondo con el ánimo de convertirse en uno de los documentos más completos en esta temática. Para conseguir este trabajo el texto presentado toma como referencia trabajo previo de los autores, complementándolo y orientándolo para hacer su lectura más asequible. Algunos de estos trabajos son: el tercer capítulo del libro "Aplicaciones Criptográficas" publicado el Departamento de Publicaciones de la Escuela Universitaria de Informática de la Universidad Politécnica de Madrid, España, y el curso El algoritmo RSA que puede consultar en el MOOC Crypt4you.

El técnico o experto en seguridad tendrá especial interés por el sistema RSA, aunque le venga muy bien recordar sus inicios en la criptografía como texto de amena lectura, plagado de recuerdos y, por su parte, el lector no experto en estos temas criptológicos pero sí interesado, tal vez le atraiga inicialmente la criptografía clásica por su sencillez y sentido histórico, a quien con este libro queremos incentivarle a dar el salto a la criptografía moderna, que verá no es tan difícil de entender como pueda pensarse en un principio. Sin duda, encontrará un conocimiento más inmutable para gente con un alma más matemática, otros serán interesantes para personas con una sensibilidad más práctica e ingeniera. Tal vez la mezcla de todo constituya un buen aderezo para su formación.

Pero quizás todo esto no sea suficiente, porque si bien la información es poder, más cierto es que la inteligencia gestiona sutilmente la anticipación. Precisamente éste es el objetivo de este libro, el deseo de que gestione su inteligencia adecuadamente, escudriñe la base de la criptografía en las comunicaciones mundiales y decida la mejor manera de protegerse.

Acompáñenos en este camino, disfrute la píldora roja, sea un poco más fuerte, un poco más ágil, un poco más rápido. De paso, si el conocimiento de este libro le ayuda a hacer un mundo mejor, enhorabuena, habrá encontrado el camino más provechoso para utilizar la criptografía con un buen fin.

Dr. Alfonso Muñoz

Dr. Jorge Ramió

Capítulo I
Introducción

La comunicación es uno de los símbolos más brillantes del siglo XX. Su ideal, en acercar los valores y las culturas, ha liberado a los hombres de los obstáculos ancestrales de tiempo y espacio, compensando los horrores y las barbaries de nuestra época. Su avance ha acompañado a los combates por la libertad, los derechos humanos y la democracia.

Hoy día, la sociedad del conocimiento en la que estamos inmersos extiende el potencial de las tecnologías de comunicación a todas las facetas de la vida cotidiana. La idea clásica de que la religión es el opio del pueblo, hoy día es sustituida por la información, su adquisición supone poder, y ésta se cree que conduce a la libertad. El poder de la información es tal que numerosas entidades a lo largo de la Historia han deseado e intentado restringirla o manipularla. De hecho, el espionaje de las comunicaciones ha constituido, tanto en los períodos de guerra como en los tiempos de paz, un valioso instrumento para conocer las actividades e intenciones de otros grupos de personas, y así luego poder actuar en consecuencia.

1.1 Desde la criptografía simétrica a la criptografía cuántica

El ser humano siempre ha tenido secretos de muy diversa índole y ha buscado mecanismos para mantenerlos fuera del alcance de miradas indiscretas, especialmente si la información es transmitida por un canal inseguro, que posibilite que pueda ser curioseada y/o modificada. La evolución de todos los mecanismos y técnicas que intentan solucionar este problema es lo que se denomina hoy en día como la ciencia de la criptología, que está compuesta por sus dos ramas, la criptografía y el criptoanálisis.

La ciencia de la criptología se puede englobar en dos grandes épocas: criptología clásica y criptología moderna. La criptología clásica comprende todas aquellas técnicas de escritura secreta hasta mediados del siglo XX. Estas técnicas de cifra se agrupaban en métodos de transposición y métodos de sustitución. La transposición consiste en colocar-combinar-reordenar la información de un mensaje de formas distintas a la original, mientras que la sustitución establece mecanismos que consisten en la sustitución de caracteres del alfabeto empleado por otros símbolos, típicamente mediante sustitución monoalfabética o polialfabética.

Con el paso de los siglos la ciencia de la criptología fue adquiriendo consistencia. Se conocían multitud de algoritmos de cifrado y métodos de criptoanálisis. Una nueva criptografía estaba a punto de saltar a escena, con los mejores avances de los siglos anteriores, entre ellos las ideas planteadas en 1883 por el lingüista holandés *Auguste Kerckhoffs* en su libro *La cryptographie militaire.* Estas ideas, más tarde conocidas como los principios de *Kerckhoffs*, serían de utilidad en el diseño de los criptosistemas modernos.

Los dos principios más importantes son:

- La seguridad de un criptosistema no debe depender de mantener secreto el algoritmo de cifrado. La seguridad sólo debe depender de mantener la clave de cifrado en secreto.
- Si el criptosistema no es teóricamente irrompible, al menos debe serlo en la práctica.

El camino hacia una nueva filosofía criptográfica ya había comenzado, y se fue robusteciendo con una serie de artículos que establecieron definitivamente la base de la nueva criptografía, la criptografía moderna, y que se extiende hasta nuestros días.

Existen dos momentos claves en el siglo XX para la evolución futura de la criptografía y la protección de comunicaciones. Uno de ellos fue en la década de los 40 del siglo XX con la publicación de dos artículos fundamentales que sentarían las bases de la teoría de la información: *A Mathematical Theory of Communication*, en 1948, y *Communication Theory of Secrecy Systems,* en 1949, desarrollados por *Claude Shannon*.

Los artículos de *Shannon* propusieron dos técnicas de cifrado en criptosistemas de clave secreta, que resumían los mecanismos anteriores de la historia, a las que llamó difusión y confusión. Por un lado, la difusión sería la técnica que permitiría dispersar las propiedades estadísticas inherentes al lenguaje en el texto en claro sobre el criptograma, por ejemplo, mediante permutaciones o transposiciones. Por otro lado, la técnica de confusión, permitiría generar confusión, caos, mezcla en el resultado cifrado, de tal forma que la dependencia entre texto en claro, clave y criptograma sería lo más compleja posible e impediría romper el algoritmo (propone aplicar la técnica de sustitución). Ahora más que nunca la criptografía se convertiría en el refugio de los matemáticos, el lugar perfecto en el cual aplicar numerosas teorías, teniendo en cuenta los principios de *Kerckhoffs*. Todos estos avances contribuirían al desarrollo de cifradores de flujo y cifradores de bloque.

A finales de la década de los 70 y en la década de los 80 se producirían los avances conceptuales más notorios que marcarían muchas de las tendencias criptográficas en las décadas posteriores. Posiblemente, el salto cualitativo más importante en la historia de la criptografía fue gracias al artículo *New directions in cryptography*, publicado en 1976 por *Bailey Whitfield Diffie* y *Martin Hellman*, que establecía el concepto de criptografía asimétrica o clave pública, en la que cada participante en una comunicación secreta disponía de dos claves, una pública y otra privada. Cualquier emisor podía comunicarse con un destinatario conociendo exclusivamente su clave pública, en tanto que sólo el destinatario podía descifrar la comunicación cifrada, dado que sólo él conocía su clave privada.

Fue en esta época cuando se abrió el camino al uso de funciones unidireccionales fáciles de computar en una dirección pero muy complejas computacionalmente de invertir sin una trampa, una pista a modo de clave de sistema. Esto abrió un gran potencial para el desarrollo de protocolos criptográficos en las redes de comunicación y para dar una solución práctica al problema de la distribución de claves. En la década de los 80, el avance en los principios de los algoritmos de curvas elípticas y en las ideas de la actual criptografía cuántica marcarían los sistemas actuales de protección de comunicaciones digitales.

Desde finales de los 80 hasta nuestros días surgirían multitud protocolos, algoritmos y herramientas utilizando todos estos principios. La criptografía se convertiría en un recurso obligatorio para garantizar la confidencialidad, integridad, autenticidad y no repudio de las comunicaciones en el mundo globalizado en el cual vivimos.

■■■

1.2 El problema de la distribución de claves. Criptografía pública y PKI

La criptografía simétrica fue de gran utilidad en entornos militares, diplomáticos y políticos, hasta la expansión de las redes de comunicaciones e Internet y al uso de la criptografía en las comunicaciones civiles, en el cual influyó notoriamente la publicación de la herramienta PGP por *Phil Zimmermann* a comienzos de la década de los 90.

Antes de los avances significativos citados en el siglo XX muchos de los algoritmos criptográficos simétricos basaban gran parte de su seguridad en la oscuridad. Emisor y receptor de una comunicación utilizaban un algoritmo (sistema de cifra) secreto conocido entre ambos, si un atacante conocía el algoritmo la comunicación secreta sería revelada. Mucho antes de que se documentaran los principios de *Kerckhoffs*, diferentes algoritmos criptográficos observaron que la seguridad por oscuridad tarde o temprano no sería suficiente y empezaron a manejar claves criptográficas. Una clave criptográfica sería una pequeña información secreta compartida entre emisor y receptor, el sistema de cifra ya no tendría por qué ser secreto. Esta idea brillante, que es la base de cualquier sistema moderno, tiene varios inconvenientes intrínsecos, entre ellos, cómo proteger la clave criptográfica de miradas indiscretas y cómo distribuirla entre las partes de la comunicación de una manera segura.

Hoy día existen propuestas, bajo ciertas suposiciones, para intercambiar claves mediante el uso de protocolos telemáticos que utilizan cifra simétrica, ejemplos son los centros de distribución de claves (KDC) o el protocolo *ZimmKerberos*, pero hasta el advenimiento de la informática y la telemática estos procedimientos, de haber existido, hubieran sido poco prácticos.

En realidad hasta la invención de la criptografía pública en la década de los 70 en el siglo XX un emisor y un receptor debían comunicar la/s clave/s criptográficas por algún canal lo "más seguro y rebuscado posible": intercambiarla personalmente entre mensajeros de confianza, hacer uso de esteganografía ocultando la clave en algún medio, utilizar la valija diplomática que se suponía inviolable, etc. En ciertas ocasiones esas claves podrían derivar en nuevas claves para minimizar su intercambio. Es fácil comprender la dificultad que existe en la criptografía simétrica para distribuir claves. Si sólo existiera este tipo de criptografía no hubiera sido posible la existencia de mecanismos seguros en Internet de una manera escalable y mucho menos el comercio electrónico.

Es precisamente en este punto donde la criptografía pública tiene un especial interés ya que permite solucionar el problema de la distribución de claves a través de un canal inseguro. La clave pública de un usuario destino (conocida por todos) puede ser utilizada para cifrar una clave, típicamente una clave de sesión simétrica; sólo el destinatario podrá recuperarla porque sólo él conoce su clave privada que deshace la cifra. No obstante, la criptografía pública no es una solución perfecta ya que todavía sufre el problema de garantizar la autoría de la clave pública del receptor/destino de una comunicación.

El ataque más común en criptografía de clave pública es un ataque del tipo hombre en el medio, *man in the middle*, es decir, un tercero (atacante) hace creer al emisor de una comunicación que la clave pública del atacante es la del destinatario. Si consigue tal engaño, que en la práctica es muy sencillo de conseguir con herramientas de auditoría clásicas, se interpone en la comunicación de forma transparente pudiendo obtener toda la información a proteger en claro.

Los pasos del ataque *man in the middle* son:

- El emisor usa la clave pública del atacante para enviar la clave de sesión, pensando que la clave pública es la del destino, por ejemplo, la página web de un banco.
- El atacante descifra con su clave privada y obtiene la clave de sesión, a continuación utiliza la clave pública del destino para enviarle la clave. De esta forma la interceptación es transparente.

Por tanto, la criptografía pública soluciona el problema de la distribución de claves pero se deben añadir mecanismos para garantizar la identidad de las claves públicas de las partes de una comunicación.

Sin recurrir a grandes infraestructuras, se puede recurrir a mecanismos más o menos a medida que tienen varios inconvenientes, entre ellos su falta de escalabilidad. Son muy famosas propuestas basadas en anillos de confianza como, por ejemplo, las firmas PGP entre usuarios para generar anillos de confianza. Imagine el siguiente escenario: Pepe se fía de Juan, luego certifica la clave pública de Juan. Tomás se fía de Pepe, luego se fía de lo que certifica Pepe y por tanto se fía de la identidad de la clave pública de Juan. Es lo que en lenguaje coloquial y popular se representa por la frase "los amigos de tus amigos son mi amigos", nada aconsejable lógicamente en un entorno de documentos cifrados, secretos, etc.

Independientemente de estas propuestas, en la actualidad no se puede hablar de criptografía de clave pública sin mencionar a las infraestructuras de clave pública (PKIs), las autoridades de certificación o tercera parte confianza (CA) y los certificados digitales. En pocas palabras verificar la identidad de una clave pública se soluciona incorporando el concepto de tercera parte de confianza. Una tercera parte de confianza es un elemento intermedio que actúa como juez en esa verificación gracias a que mediante el uso de criptografía pública la tercera parte de confianza certificará/firmará la clave pública de cada comunicante.

Un lector aventajado observará rápidamente que una tercera parte de confianza no es suficiente para solucionar el problema ya que a su vez alguien podría suplantar a esa parte. Esto se intenta solucionar mediante mecanismos administrativos ("cualquiera" no debería ser una tercera parte de confianza de aceptación masiva) y mediante mecanismos técnicos. Técnicamente el emisor y receptor de una comunicación debe tener, por defecto, la clave pública de la tercera parte de confianza para interactuar con ella y verificar las claves públicas de posibles comunicaciones origen-destino.

Un ejemplo clásico de este funcionamiento en el mundo real son los navegadores web y la comunicación vía protocolo https con servidores web. Los navegadores tienen instalados por defecto las claves públicas de las CA más famosas (terceras partes de confianza). Cuando el navegador recibe el certificado público del servidor (su clave pública) firmado por una CA "famosa" puede verificar que la clave pública es de quien dice ser y, por tanto, iniciar un protocolo para el intercambio de una clave de sesión para cifrar los datos en tránsito[1].

Exceptuando diferentes algoritmos matemáticos de clave pública, entre ellos los de curvas elípticas, no se han publicado aportaciones conceptuales significativas para solventar el problema de

1 En próximos capítulos se destacarán múltiples consideraciones a tener en cuenta sobre la seguridad del funcionamiento e implementación de las CAs en el mundo real y los certificados digitales.

la distribución de claves criptográficas desde la década de los 80. Es posible que el lector haya leído sobre una posible solución "novedosa" (sus inicios se remontan a la década de los 80) a la distribución de claves mediante el uso de criptografía cuántica, sin embargo esto debe ser matizado.

La criptografía cuántica permite, mediante una serie de protocolos (BB84, B91, E91...), intercambiar una cantidad determinada de bits aleatorios entre emisor y receptor que se pueden utilizar como clave o semilla para generar una nueva clave. Este intercambio se basa en el envío de fotones y en ciertas propiedades físicas no violables, como el teorema de no-clonación. Dejando de lado problemas en implementaciones reales, teóricamente es una propuesta robusta y la propiedad de no poder duplicar los fotones enviados (teorema de no-clonación) permite mediante procedimientos estadísticos detectar el "pinchazo" de un canal, detectando al "espía". La criptografía cuántica no soluciona el problema que se soluciona con las PKIs. La criptografía cuántica se emplea en comunicaciones punto a punto, y dichos puntos deben están identificados previamente para evitar un ataque tipo man in the middle.

Por tanto, la criptografía pública es la tecnología actual para facilitar masivamente el intercambio de claves criptográficas, el cifrado de datos en redes de telecomunicación y, como se verá posteriormente, la posibilidad del firmado digital. Entre los múltiples algoritmos dentro de este tipo de criptografía hay uno cuya importancia sobrepasa a todos los demás, el algoritmo RSA. Este algoritmo es utilizado masivamente en Internet y conocer su diseño y seguridad permite adentrarse en la protección de las comunicaciones mundiales en la actualidad.

En el interés de formar a los lectores en este aspecto surge este libro. En los próximos capítulos se intenta dar luz sobre todo esto.

Capítulo II
Sistemas de cifra clásica y su evolución a criptosistemas simétricos modernos

Este capítulo sintetiza de una manera sistematizada las técnicas y artilugios que posiblemente para un lector aventajado no dejen de ser un mero *entretenimiento cultural* comparado con la importancia de la criptografía o algoritmos modernos. No obstante, los sistemas clásicos son especialmente sencillos y didácticos, lo que los convierte en idóneos para comprender la evolución de esta ciencia y fortalecer conceptos. Además, no existe mucha bibliografía fácilmente accesible sobre esta cuestión en español y siempre es recomendable saber qué se hizo en el pasado para poder valorar lo que se tiene en el presente y vislumbrar de alguna manera el futuro.

Sirva entonces este capítulo como compendio relativamente amplio de los sistemas de cifra clásica más famosos, desde sus inicios hasta mediados del siglo XX, en que el advenimiento de la informática y la representación de la información en formato binario producen un importante cambio en la criptografía. Este repaso a los sistemas de cifra de antaño está presentado bajo el prisma de una aportación didáctica que ayude a entender la criptografía actual, así como las matemáticas que se esconden detrás de aquellos antiguos sistemas de cifra, que aunque le parezca increíble no distan mucho de los que se usan en la actualidad si se trata de sistemas de cifra simétrica.

2.1. Introducción

¿Qué se entiende por criptosistemas clásicos? En el capítulo anterior se comentaba que los sistemas de cifra podían clasificarse de varias formas, siendo la más aceptada aquella que toma en cuenta la característica del secreto de la clave, dando lugar a criptosistemas de clave secreta y criptosistemas de clave pública. Ahora bien, la criptología tal y como hoy en día se concibe, una técnica de enmascaramiento de la información estrechamente unida al mundo de la informática, las redes de ordenadores y las autopistas de la información, poco tiene que ver con aquella asociada a fascinantes máquinas de cifrar, que adquirieron gran fama tras su uso en la Segunda Guerra Mundial y más aún, remontándonos a siglos pasados, con los métodos, técnicas y artilugios utilizados por emperadores, gobernantes, militares y en general diversas civilizaciones para mantener sus secretos a buen recaudo.

En aquellos tiempos, el mundo de la criptología estaba vinculado directamente con el poder fáctico, ligado a secretos de estado, asuntos militares, de espionaje y diplomáticos, en todo caso siempre seguido de una aureola de misterio y que incluso salta a la literatura de ficción en el cuento "El escarabajo de oro" de *Edgar Allan Poe*, publicado en 1843 en "Dollar Newspaper". Se trata de un relato de aventuras cuyo eje principal gira en torno al criptoanálisis de un conjunto de caracteres extraños que aparecen en un pergamino cifrado y cuyo texto esconde el lugar exacto donde se

encuentra enterrado el valioso tesoro de un pirata de nombre *Kidd*. El sistema de cifra es uno de los más simples, el denominado monoalfabético por sustitución con alfabeto mixto, de forma que nuestro protagonista *William Legrand* no tiene más que aplicar las estadísticas del lenguaje, que cien años después de este libro estudiará a fondo *Claude Shannon*, alguna que otra suposición sobre formación de palabras y una pizca de intuición para hacer corresponder los signos del enigmático criptograma con letras del alfabeto y así descifrar el mencionado pergamino. Si lo desea, es una lectura recomendable.

A comienzos del siglo XX el uso de la criptografía en las transmisiones de mensajes cobra una importancia inusitada por los tiempos que corrían (I y II Guerra Mundial), originando esto un gran auge tanto de las técnicas como de las máquinas de cifrar (*Enigma*, *Hagelin*, *Purple*, etc.). Ejemplos famosos se encuentran a pares. El 17 de enero de 1917 *William Montgomery*, criptoanalista de la sección diplomática de la famosa Habitación 40 del Almirantazgo de la Marina Británica en Londres, intercepta un telegrama lleno de códigos[1] que el Ministro de Relaciones Exteriores alemán *Arthur Zimmermann* envía a su embajador en los Estados Unidos. Tras romper los códigos, descubren atónitos que entre otras cosas el mensaje anunciaba la guerra con los Estados Unidos. Con ello los Estados Unidos entran en la confrontación mundial y ayudan a los aliados a ganar la guerra. Según palabras de *David Kahn*, autor de una de las obras más completas sobre historia de la criptografía, "… nunca un único criptoanálisis ha tenido tan enormes consecuencias". De hecho, el descubrimiento de este secreto cambió el rumbo de la historia.

Y no es el único caso. Otro ejemplo histórico cae en plena II Guerra Mundial. El 7 de diciembre de 1941, la radio de la estación naval de Bainbridge Island, cerca de Seattle en los Estados Unidos, intercepta un mensaje de solamente 9 minutos desde Tokio a la Embajada Japonesa en los Estados Unidos. El radiotelegrama estaba cifrado con una máquina que los norteamericanos llamaron *Purple*, cuyo código fue roto por *William Friedman*, quizás uno de los criptólogos más importantes de la historia, y un grupo de criptoanalistas. Si bien es cierto que ello no pudo evitar el ataque de los japoneses a Pearl Harbor, el esfuerzo realizado por todos en la destrucción de tales códigos jugó después un papel fundamental y marcó la derrota del pueblo nipón así como el fin de la guerra.

No obstante, a pesar del importante papel que la criptografía desempeñó en las dos grandes confrontaciones mundiales, se puede afirmar que la historia de la criptografía clásica abarca desde el siglo V a. C. con la aparición en un pueblo griego de un sistema de cifra llamado escítala, hasta los años de la posguerra, es decir, hasta la mitad del siglo XX. El adjetivo de clásica, en contraposición al de criptosistemas modernos, se debe tanto a las técnicas utilizadas en las primeras, básicamente operaciones lineales de sustitución y transposición de caracteres, con o sin clave, pero siempre unido al concepto de clave secreta, como al uso de máquinas dedicadas a la cifra. En muchos casos el sistema de cifra era también un secreto guardado entre los interlocutores.

Por el contrario, los sistemas modernos hacen uso, además de lo anterior, de algunas propiedades matemáticas como, por ejemplo, la dificultad del cálculo del logaritmo discreto o el problema de la factorización de grandes números en los sistemas llamados de clave pública o asimétricos, unido

1 Se usa la palabra código porque así aparece comúnmente en la bibliografía relacionada con los sistemas de cifra clásica, aunque lo correcto es utilizar la palabra cifra pues cifrar no es lo mismo que codificar. Lo primero es dinámico y puede cambiar su resultado en función de que se use una u otra clave, mientras que codificar es una acción estática, el código una vez establecido es uno solo, inamovible y común para todos: código Morse, código ASCII, etc.

esto a la representación binaria de la información. No obstante, muchos sistemas modernos y que en la actualidad se siguen utilizando, como los algoritmos de clave secreta Triple DES y AES, se basan en conceptos que pueden denominarse clásicos como son los de transposición y sustitución con una clave privada, si bien en estos sistemas la operación se realiza sobre una cadena de bits y no sobre caracteres.

Como se ha comentado, muchos de los criptosistemas clásicos, en particular aquellos que transforman el mensaje en claro aplicando técnicas de sustitución y transposición, basaban su seguridad principalmente en el secreto de la transformación o algoritmo de cifra. Es ésta también una diferencia fundamental con respecto a los sistemas modernos, en los que el algoritmo se hace público puesto que la fortaleza del sistema reside en la imposibilidad computacional de "romper" una clave secreta. Observe que el hecho de hacer público el algoritmo de cifra permite al criptólogo evaluar la calidad del software desarrollado, puesto que éste será estudiado por la comunidad científica intentando buscar algún defecto, una puerta falsa, una rutina innecesaria, una codificación no depurada, etc.

De todos los sistemas clásicos, cuya diversidad es enorme, en este capítulo se analizarán solamente algunos; los más conocidos y que de alguna forma servirán como apoyo para profundizar y aplicar algunos conceptos que deberían conocerse sobre cifra moderna en general y el uso de las matemáticas discretas en esos sistemas. Como comprobará el lector, se proporciona una extensa documentación sobre mecanismos de criptoanálisis de los algoritmos. Si usted es un apasionado de esta rama de la criptología se recomienda encarecidamente la lectura de los textos clásicos de criptoanálisis militar de *William F. Friedman* citados en la bibliografía.

2.2 Alfabetos y características del lenguaje

2.2.1. Alfabetos de cifrado

En la mayoría de los cifradores clásicos se utiliza como alfabeto de cifrado el mismo alfabeto del texto en claro. Para poder aplicar las operaciones de transformación se asocia a cada letra del alfabeto un número de forma que a la letra A le corresponde, por ejemplo, el valor 0, a la letra B el 1, etc. De esta manera, si se trata del castellano, pueden definirse varios tipos de alfabetos, como por ejemplo:

Alfabeto 1: Letras mayúsculas: aritmética módulo 27.

Alfabeto 2: Letras mayúsculas con dígitos 0-9: aritmética módulo 37.

Alfabeto 3: Letras mayúsculas y minúsculas: aritmética módulo 54.

Alfabeto 4: Letras mayúsculas y minúsculas además acentuadas: aritmética módulo 59.

Alfabeto 6: Letras mayúsculas, minúsculas además acentuadas y dígitos: aritmética módulo 69.

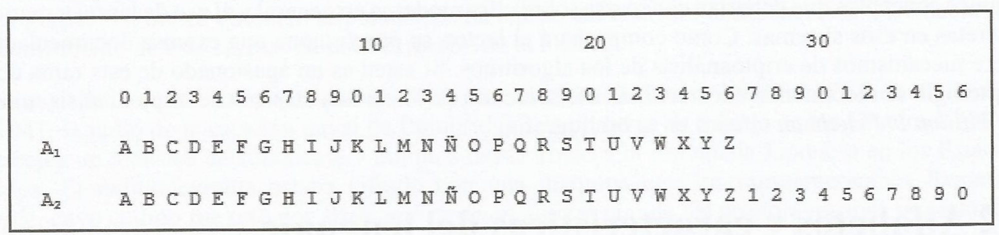
Alfabeto 7: Todos los caracteres imprimibles ASCII: aritmética módulo 224.

En los seis primeros casos no se tiene en cuenta el carácter del espacio en blanco (valor ASCII 32) puesto que ello entregaría en muchos cifradores clásicos una apreciable pista al hipotético criptoanalista. Tenga en cuenta que para un texto en castellano en el que el alfabeto considerado sea

el de 27 letras más el espacio en blanco, este último presenta una frecuencia de ocurrencia de casi un 20%, siguiéndole muy por detrás las letras E y A, con valores en el orden del 10%. No obstante, si se elimina este carácter y se cifran los mensajes solamente con las 27 letras del alfabeto, la letra E presenta una frecuencia de aproximadamente un 13%, la letra A se alza por encima del 10% y los demás caracteres siguen una distribución característica que será tratada en el apartado siguiente.

En cuanto al último alfabeto presentado, hay 224 caracteres imprimibles en ASCII, desde el valor 32 al 255, contando claro está con el espacio en blanco, o bien 223 sin éste. Si se utiliza como alfabeto de cifrado el código ASCII, hay que tener especial cuidado con los caracteres no imprimibles, caracteres especiales como los de salto de línea, fin de archivo, etc., que luego podrían no ser recuperables. Es decir, la operación de cifrado y descifrado debe considerar estas condiciones de forma que sólo se transmitan los caracteres que pueden imprimirse y no incluir caracteres extraños en el criptograma. Evidentemente esto sólo tiene sentido en este tipo de cifradores orientados a caracteres en donde para nada se habla de bits. La cifra moderna es digital y, por tanto, este problema no existe.

La siguiente figura muestra la correspondencia del alfabeto de texto en claro con números para el alfabeto 1 y el alfabeto 2. Se dirá entonces que las operaciones de transformación se realizan en módulo 27 y módulo 37, respectivamente.



											10										20										30						
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
A_1	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z										
A_2	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	1	2	3	4	5	6	7	8	9	0

Figura 1. Correspondencia de números en alfabetos.

Establecer una correspondencia numérica de los caracteres del alfabeto permitirá realizar operaciones matemáticas de desplazamiento (adición) y decimación (multiplicación)[2] a los caracteres del texto en claro, de forma que se puedan aplicar todas las propiedades de la aritmética. Por ejemplo, siguiendo la tabla de correspondencia del alfabeto A_1 y recordando conceptos de congruencia que podrá repasar en el Apéndice correspondiente de este libro, se puede formar la palabra TARZAN a partir de las siguientes operaciones matemáticas, sin relación alguna entre ellas. Obviamente hay otra gran cantidad de operaciones y valores para llegar al mismo resultado.

T	X - E = 24 - 4 = 20 = T
A	4*A = 4*0 = 0 = A
R	H + L = 7+11 = 18 = R
Z	8*K= 8*10 = 80 (mayor que 27) ⇒ 80 - 2*27 = 26 = Z
A	J + R = 9 + 18 = 27 (mayor que 27) ⇒ 27 - 1*27 = 0 = A
N	T + T = 20 + 20 = 40 (mayor que 27) ⇒ 40 - 1*27 = 13 = N

Observe que los resultados se han reducido módulo 27, esto es dividiendo el resultado por este valor las veces necesarias y quedándose con el resto o residuo de dicha operación. Luego, cualquier

2 Aunque la palabra "decimación" no existe, se utilizará aquí para indicar cómo se recorre un módulo dando saltos, en donde el tamaño de ese salto corresponde al valor indicado como "decimación".

operación tanto de desplazamiento como de decimación, será representada en el mismo cuerpo o módulo.

Para que estas transformaciones puedan ser aplicadas en criptografía y pueda recuperarse el texto en claro a partir del texto cifrado o criptograma, deberá cumplirse que exista el inverso. En el caso de la suma siempre existe el inverso, no así para la multiplicación. Si estas últimas afirmaciones le resultan algo extrañas, no se preocupe; como ya se ha comentado existe un apéndice dedicado a la aritmética modular que puede consultar si es necesario.

2.2.2 Estadísticas del lenguaje

El lenguaje castellano presenta una gran redundancia. Esto significa que en algunos criptosistemas (básicamente los de tipo clásico orientados al cifrado de caracteres) se puede aplicar esta característica para criptoanalizar textos cifrados. De hecho, lo primero que se plantea todo criptoanalista es suponer que el cifrado podría ser de tipo básico y, por lo tanto, debería intentarse el ataque a partir de las estadísticas del lenguaje. En lo que concierne a los cifradores clásicos, éstos se dividen en monoalfabéticos y polialfabéticos, en tanto se utilice un único alfabeto para cifrar o más de uno. En tales casos, el análisis de las frecuencias relativas de aparición de los caracteres en el criptograma podría indicarnos si se trata de uno u otro tipo de cifra.

Aunque los sistemas clásicos estén en desuso, no por ello deben ser pasados por alto por el criptoanalista. En realidad sería bastante poco agradable perder horas de esfuerzo en la intención de romper una cifra, suponiendo de antemano que el criptosistema en cuestión empleado es de los denominados modernos, para luego caer en la cuenta que aquel complicado enigma se trataba simplemente de un cifrado elemental, que podría romperse fácilmente con herramientas básicas. No quedaría muy bien ante sus superiores. Por lo tanto, la primera acción que realizará todo criptoanalista será la de contabilizar los caracteres que aparecen en el criptograma para obtener alguna información sobre el tipo de cifra, monoalfabético o polialfabético, e intentar aplicar las técnicas que se describirán más adelante para romper dicha cifra. Si esto no entrega los resultados esperados, buscará otros caminos, yendo como es lógico siempre desde la dificultad menor a la mayor.

En la siguiente tabla se incluyen las frecuencias relativas de monogramas en el lenguaje castellano módulo 27 considerando sólo las letras mayúsculas. Estos datos nos permiten formar tres grupos de frecuencias relativas: uno de alta frecuencia, otro de frecuencia media y un tercero de frecuencia baja.

Frecuencia Alta	Frecuencia Media	Frecuencia Baja
E-13,11	C-4,85	Y-0,79
A-10,60	L-4,42	Q-0,74
S-8,47	U-4,34	H-0,60
O-8,23	M-3,11	Z-0,26
I-7,16	P-2,71	J-0,25
N-7,14	G-1,40	X-0,15

Frecuencia Alta	Frecuencia Media	Frecuencia Baja
R-6,95	B-1,16	W-0,12
D-5,87	F-1,13	K-0,11
T-5,40	V-0,82	Ñ-0,10

Tabla 1. Clasificación de frecuencias relativas expresadas en tanto por ciento de caracteres del lenguaje español módulo 27

En la tabla anterior, se ha considerado como Frecuencia Alta un valor mayor que el 5 por ciento y Frecuencia Baja un valor similar o menor que un 1 por ciento. Observe que mezclando las letras de alta frecuencia se puede formar la palabra ESTIRANDO. Más adelante volverán a considerarse estos nueve caracteres cuando se aborden las técnicas de criptoanálisis.

Como se ha indicado el texto del que se obtienen estas frecuencias no es necesario que sea de gran tamaño, basta unas cuantas páginas para que esos datos ya se mantengan constantes. Sin embargo, dependiendo del tipo de documento analizado aparecerán ligeras diferencias, aunque se puede concluir que los valores se mantienen en el rango indicado. Esto quiere decir que es posible considerar, por ejemplo, la letra L con más peso que la D, incluir en la zona de alta frecuencia la letra C en vez de la letra T, etc. Piense por ejemplo en algún documento que contenga información sobre comercialización de un determinado producto agrícola; es posible que la letra K tenga una contribución mayor por el hecho de que aparezca muchas veces la palabra kilo; lo mismo en un informe médico de radiología, donde la letra X puede tener un mayor peso que el aquí indicado. No obstante, el estudio estadístico de la frecuencia de caracteres tendrá su utilidad sólo en el criptoanálisis de sistemas clásicos por sustitución, en donde se supondrá que los mensajes a cifrar tratarán siempre de textos comunes. Es más, en la mayoría de los casos tales mensajes contienen solamente caracteres alfabéticos y no del tipo alfanuméricos.

La redundancia[3] del lenguaje no sólo nos dice que existen letras más frecuentes que otras. También nos indica la existencia de digramas comunes, trigramas, poligramas, y en general palabras de mayor uso que otras como se muestra en este ejemplo.

Ejemplo: Dado el siguiente texto clásico se pide encontrar:

a) Las frecuencias relativas de monogramas.

b) Los 9 monogramas de mayor frecuencia.

c) La frecuencia relativa de digramas.

d) Los tres digramas más frecuentes.

"En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda. El resto de ella concluían sayo de velarte, calzas de velludo para las fiestas, con sus pantuflos de lo mismo, y los días de entre semana se honraba con su vellorí más fino. Tenía en su casa una ama que pasaba de los cuarenta, y una sobrina que no llegaba a los veinte, y un mozo de campo y plaza, que así ensillaba

3 Términos como redundancia del lenguaje, entropía, ratio y distancia unicidad que verá en este capítulo corresponden a conceptos de Teoría de la Información que encontrará en el Apéndice correspondiente.

el rocín como tomaba la podadera. Frisaba la edad de nuestro hidalgo con los cincuenta años; era de complexión recia, seco de carnes, enjuto de rostro, gran madrugador y amigo de la caza. Quieren decir que tenía el sobrenombre de Quijada, o Quesada".

Solución: a) Las frecuencias relativas de monogramas módulo 27 en % para este trozo de texto son:

A 14,38	J 0,41	R 5,75
B 1,64	K 0,00	S 7,53
C 4,38	L 6,99	T 2,88
D 5,75	M 3,15	U 4,93
E 11,37	N 7,53	V 1,10
F 0,68	Ñ 0,00	W 0,00
G 1,92	O 9,73	X 0,14
H 1,10	P 1,51	Y 1,23
I 3,70	Q 1,51	Z 0,68

b) Los nueve monogramas más frecuentes en el texto son: A, D, E, L, N, O, R, S, U

c) Los valores absolutos de frecuencia de digramas en el texto son:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	5	6	7	13	6	1	0	0	0	0	0	13	5	9	0	3	1	5	10	14	0	2	0	0	0	2	2
B	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0
C	7	0	0	0	0	0	0	3	4	0	0	1	0	3	0	11	0	0	0	0	0	3	0	0	0	0	0
D	10	0	0	1	22	0	0	0	1	0	0	0	0	0	0	5	0	0	1	0	0	2	0	0	0	0	0
E	3	1	9	3	2	0	1	1	1	1	0	14	2	18	0	0	1	1	8	11	1	0	3	0	1	1	0
F	0	0	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
G	5	0	0	0	0	0	0	0	0	0	0	0	0	1	0	6	0	0	1	0	0	1	0	0	0	0	0
H	3	0	0	0	1	0	0	0	2	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
I	1	0	1	3	6	0	2	0	0	1	0	2	0	7	0	0	0	0	1	2	0	0	1	0	0	0	0
J	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
K	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	14	0	0	0	4	0	5	0	0	0	0	7	0	0	0	13	1	0	2	1	0	3	0	0	0	0	1
M	6	2	0	0	1	0	0	0	4	0	0	0	0	0	0	3	3	0	0	3	0	1	0	0	0	0	0
N	8	0	4	2	3	1	1	1	0	1	0	4	2	0	0	8	1	0	2	6	8	2	0	0	0	0	1
Ñ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
O	2	2	5	9	0	0	1	2	0	0	0	2	9	7	0	0	1	3	5	16	3	0	0	0	0	3	1
P	5	0	0	0	0	0	0	0	1	0	0	2	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0
R	7	0	0	2	10	0	1	0	2	0	0	0	2	3	0	8	0	1	1	0	2	2	0	0	0	1	0
S	7	1	4	8	6	2	0	0	1	0	0	3	2	1	0	2	2	0	0	1	6	5	3	0	0	1	0
T	3	0	0	0	6	0	0	0	3	0	0	0	0	0	0	4	0	0	4	0	0	1	0	0	0	0	0
U	3	0	2	1	11	1	2	1	3	0	0	0	1	6	0	0	0	0	1	1	1	0	1	0	0	1	0

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
V	2	0	0	0	4	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	2	1	1	0	0	0	2	0	0	0	0	0
Z	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

d) Los tres digramas más frecuentes del texto son DE con 22 apariciones, EN con 18 y OS que aparece 16 veces.

En el ejemplo anterior, a pesar de que el texto no tiene la longitud que sería recomendable para obtener unos resultados que sean fieles a la realidad de la ratio y redundancia del lenguaje, sí deja entrever una tendencia marcada del mayor peso de algunas letras y conjunto de letras. De las 9 letras de mayor peso en este texto, 7 corresponden a la clasificación de Alta Frecuencia que se había presentado.

En cuanto a los digramas, existe una mayor dispersión como es natural porque el texto analizado es muy corto. No obstante, dos de los tres digramas más comunes del texto, DE con un 3,0 % y EN con el 2,5 %, son también los más frecuentes en el lenguaje castellano con valores de frecuencia muy similares. Como podrá suponer, esto tiene gran importancia, tanta que el estudio de frecuencias del lenguaje constituye una parte central en la mayoría de tratados de criptoanálisis de los algoritmos clásicos.

2.3 Clasificación de los criptosistemas clásicos

Los métodos clásicos son aquellos en los que se usan técnicas de sustitución y transposición aplicadas a los caracteres del texto en claro. Las técnicas criptográficas utilizadas en este caso son en su totalidad orientadas a sistemas de clave secreta, generalmente manteniendo también en secreto el algoritmo. La operación de cifra se realiza sobre caracteres alfanuméricos, por lo general alfabéticos, y en ese mismo formato y cuerpo se transmiten o almacenan. En la siguiente figura se muestra una clasificación de los sistemas de cifra clásicos, en donde se incluyen algunos cifradores típicos a modo de ejemplo. Estos sistemas de cifra se clasificarán, básicamente, en aquellos que utilizan técnicas de sustitución y aquellos que utilizan técnicas de transposición sobre los caracteres de un texto en claro.

Los cifradores por transposición utilizan la técnica de permutación de forma que los caracteres del texto se reordenan mediante un algoritmo específico. Un caso representativo de esta transformación -que será analizado más detenidamente en un apartado próximo- sería transmitir el mensaje en bloques de cinco caracteres pero reordenados (permutados) de forma que su posición en el criptograma sea, por ejemplo, 43521; es decir, el cuarto carácter del bloque en claro se transmite primero, a continuación el tercero, después el quinto, luego el segundo y, por último, el primero. Esta operación se repetirá en cada bloque de 5 caracteres del mensaje. Por lo tanto, la transposición implica que los caracteres del criptograma serán exactamente los mismos que los del texto en claro.

Ejemplo: Cifre mediante transposición de bloques de cinco caracteres el siguiente mensaje, usando la permutación 43521.

M = AL GRITO DE VIVA ZAPATA SE ARMÓ UNA GORDA

Solución: Siguiendo la permutación indicada, se obtiene:

M = ALGRI TODEV IVAZA PATAS EARMO UNAGO RDAXX

C = RGILA EDVOT ZAAVI ATSAP MROAE GAONU XAXDA

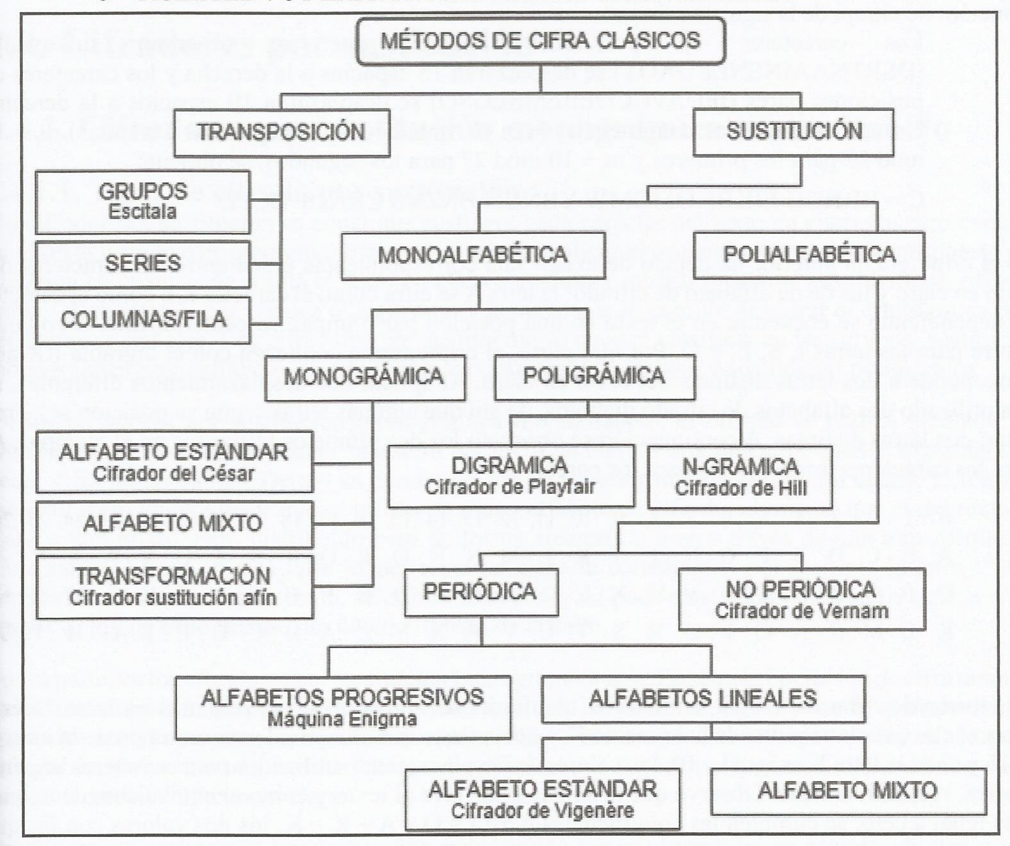


Figura 2. Clasificación de los métodos clásicos de cifra y algunos ejemplos.

Los cifradores por sustitución utilizan la técnica de modificación de cada carácter del texto en claro por otro correspondiente al alfabeto de cifrado. Si el alfabeto de cifrado es el mismo que el del mensaje o bien único, se habla entonces de cifradores monoalfabéticos; es decir, existe un único alfabeto en la operación de transformación del mensaje en criptograma. Por el contrario, si en dicha operación interviene más de un alfabeto, se dice que el cifrador es polialfabético.

¿Cómo es posible utilizar más de un alfabeto en la operación de cifrado? La respuesta es muy sencilla y se abordará a continuación. Como se verá más adelante, en el cifrador del *César* la letra A del texto en claro se cifraba siempre como la letra D; es por tanto un cifrador monoalfabético. A continuación se implementará un algoritmo sencillo para usar más de un alfabeto. Suponga que se desea diseñar un algoritmo de cifrado similar al del *César*, de forma que a los caracteres en

posiciones impares se les aplique un desplazamiento de 15 espacios a la derecha del alfabeto y a los caracteres pares un desplazamiento de 10 espacios también a la derecha según el siguiente ejemplo.

Ejemplo: Utilizando el algoritmo propuesto en el párrafo anterior, se pide cifrar el mensaje:

M = DISFRUTAN VACACIONES EN EL MES DE AGOSTO

Solución: Se cifrará de la siguiente forma:

Los caracteres en posiciones impares que se observan subrayados (DSRTNAAINSNLEDAOT) se desplazarán 15 espacios a la derecha y los caracteres en posiciones pares (IFUAVCCOEEEMSEGSO) se desplazarán 10 espacios a la derecha. Usando entonces las congruencias (vea el Apéndice de Matemáticas Discretas) $m_i + 15$ mod 27 para los primeros y $m_i + 10$ mod 27 para los segundos, se obtiene:

C = RRHOG EIKBF OMOMW YBÑHÑ BÑZVS CRÑOP DCIY.

En el criptograma anterior ha dejado de existir una correspondencia única entre los caracteres del texto en claro y los de un alfabeto de cifrado: la letra A se cifra como el carácter K o como el carácter O, dependiendo se encuentre en el texto en una posición par o impar, respectivamente. Otro tanto ocurre para las letras I, S, E, y O. Por otra parte, el criptograma comienza con el digrama RR que corresponde a dos letras distintas del texto en claro. Al aplicar dos desplazamientos diferentes, se han utilizado dos alfabetos de cifrado distintos, de ahí que algunas letras según su posición se cifren como dos letras distintas. A continuación se observan los dos alfabetos utilizados en el ejemplo; A_1 para los caracteres impares y A_2 para los pares.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
A_1	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ
A_2	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J

Esta forma de cifrar dará lugar, entre otros, al cifrador de *Vigenère* que se verá más adelante. Si éste fuera el caso, la clave utilizada habría sido K = *OK*, puesto que el equivalente numérico de la letra O es 15 y el de la letra K es igual a 10, los valores de desplazamiento utilizados para caracteres impares y pares, respectivamente. Observe que al sumarse la clave al texto y como el equivalente de la letra A es igual a cero, se cumplen las congruencias A + O = O y A + K = K, los dos valores con los que comienzan los alfabetos en cuestión.

La sustitución polialfabética puede ser periódica como en el caso del ejemplo anterior cuyo período es dos (el tamaño de la clave) o bien no periódica, cuando la clave en cuestión es tan larga como el mensaje. Tanto en el caso monoalfabético como en el polialfabético, se realiza la transformación $E_K(M)$ sobre cada uno de los caracteres del texto en claro de forma independiente; es decir, la operación se realiza carácter a carácter o lo que es lo mismo a través de monogramas. También es posible cifrar un texto en claro utilizando bloques de más de un carácter. En este caso se hablará de cifradores digrámicos que cifran el texto cada dos caracteres, trigrámicos en bloques de tres caracteres y, en general, poligrámicos. Hablar de alfabetos mixtos se refiere al uso de alfabetos que contienen caracteres distintos al propio del lenguaje, por ejemplo mediante el uso de símbolos. A continuación se puede ver un posible alfabeto de cifrado mixto.

Mi	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ci	<	>	(	)	&	%	/	$	¿	T	D	A	C	U	E	N	?	♣	♦	♥	♠	@	#	[	α	β	]

Con estos sencillos conceptos es posible adentrarse en los siguientes cifradores clásicos.

2.4. Cifradores por sustitución

2.4.1. Cifradores por sustitución monográmica monoalfabeto

2.4.1.1. Tipos de cifrado por sustitución

Un cifrador por sustitución es aquel que sustituye cada carácter del texto en claro por otro carácter en el texto cifrado o criptograma. Esta es la forma de aplicar el principio de confusión propuesto por *Shannon* en cuanto que oculta el texto en claro a cualquier intruso mediante sustituciones, excepto para el destinatario, que conoce el algoritmo y la clave que le permite descifrar el criptograma para recuperar el mensaje. Los cifradores por sustitución se pueden clasificar en tres grupos: sustitución monográmica monoalfabeto, sustitución monográmica polialfabeto y sustitución poligrámica.

En los cifradores por sustitución monográmica monoalfabeto, el cifrado se realiza mediante un algoritmo que hace corresponder una letra del texto en claro a una única letra del criptograma, es decir, cifra monogramas. De ahí su denominación de cifrador monográmico. En cuanto al término monoalfabeto, quiere decir que se utiliza un único alfabeto de cifrado, el mismo que el del texto en claro o uno mixto, pero distribuido bien de forma aleatoria o bien a través de una transformación matemática. Luego, si a la letra M del texto en claro le corresponde por ejemplo la letra V o el símbolo # del alfabeto de cifrado, se cifrará siempre igual pues existe una única equivalencia entre ambos o, lo que es lo mismo, un único alfabeto de cifrado.

Por su parte, en los cifradores por sustitución monográmica polialfabeto, la operación de cifra también se realiza carácter a carácter, es decir por monogramas. No obstante, como ya se ha mencionado en apartados anteriores, a través de una clave, algoritmo o mecanismo, se obtienen varios alfabetos de cifrado de forma que una misma letra puede cifrarse con caracteres distintos, dependiendo de su posición dentro del texto en claro.

Por último, los cifradores por sustitución poligrámica tratan el mensaje en bloques de dos o más caracteres sobre los que se aplica la transformación del criptosistema en cuestión, sustituyendo n-gramas del mensaje por n-gramas de texto cifrado.

2.4.1.2. El cifrador del César

Tal vez el cifrador monoalfabético por sustitución más famoso es el denominado Cifrador del *César*, uno de los cifradores más antiguos, atribuido al dictador romano *Julio César*. Se trata de un criptosistema en el que se aplica un desplazamiento constante igual de b caracteres sobre el texto en claro, obteniéndose así el criptograma buscado. Este tipo de cifradores, llamados genéricamente cifradores monoalfabéticos por desplazamiento puro o adición, toman en el caso del cifrador del *César* un valor de desplazamiento b igual a 3. Por lo tanto, este cifrador del *César* tendrá el alfabeto de cifrado siguiente con su equivalente numérico:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Ejemplo: Con la tabla anterior del cifrador del *César* cifre el mensaje:

M = CÉSAR EL EMPERADOR HA SIDO ASESINADO

Solución: Aplicando a cada carácter M_i su equivalente C_i, se obtiene:

C = FHVDU HÑHOS HUDGR UKDVL GRDVH VLPDG R

Este sistema de cifra sencillo, apropiado e incluso bastante ingenioso para la época, presenta un nivel de seguridad muy débil; de hecho su distancia de unicidad es muy baja. En el apartado siguiente se encontrará este valor y se demostrará que el criptoanálisis de este cifrador es verdaderamente elemental, un pasatiempo.

Cifrador del César con clave

Para aumentar la seguridad o, lo que es lo mismo, la distancia de unicidad de este cifrador, se puede incluir en el alfabeto de cifrado una clave de la siguiente forma: la clave K consiste en una palabra o frase que se escribe a partir de una posición p_0 del alfabeto en claro. Si la clave es K = *ESTOY ABURRIDO* y la posición inicial $p_0 = 3$ entonces:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Clave				E	S	T	O	Y	A	B	U	R	I	D													

Los caracteres repetidos de la clave no se escriben. Una vez escrita la clave en la posición indicada, se añaden las demás letras del alfabeto en orden y de forma modular, para obtener así el alfabeto de cifrado, como se observa en el siguiente ejemplo de alfabeto de cifrador del *César* con clave.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
	W	X	Z	E	S	T	O	Y	A	B	U	R	I	D	C	F	G	H	J	K	L	M	N	Ñ	P	Q	V

En este tipo de cifrado se deja de cumplir la condición de desplazamiento constante, característica en el sistema del *César* primario.

Ejemplo: Con el cifrador del *César* con clave cifre el siguiente mensaje usando como clave

K = POBRE CHUCHO SIBERIANO con $p_0 = 2$.

M = A PERRO FLACO TODO SON PULGAS

Solución: El alfabeto de cifrado resultante será:

M_i A B C D E F G H I J K L M N Ñ O P Q R S T U V W X Y Z

C_i Y Z P O B R E C H U S I A N D F G J K L M Ñ Q T V W X

Luego, C = YGBKK FRIYP FMFOF LFNGÑ IEYL

Como era de esperar, al tener un mayor número de combinaciones de alfabetos, existe una mayor incertidumbre respecto de la clave. La distancia de unicidad de este cifrador será mayor y, por consiguiente, el sistema presentará una mayor fortaleza.

Cifradores tipo César con alfabetos mixtos

En este tipo de cifrado se aplica también una relación única entre un elemento del alfabeto en claro y un elemento del alfabeto de cifrado, salvo que este último puede contener otros caracteres o símbolos que no pertenezcan al alfabeto del mensaje, por ejemplo el siguiente cifrador del césar con alfabeto mixto.

Mi	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ci	[	«	»	{	}	×	♥	♦	♣	♠	≠	#	@	%	&	(	)	=	>	<	0	1	2	3	4	5	]

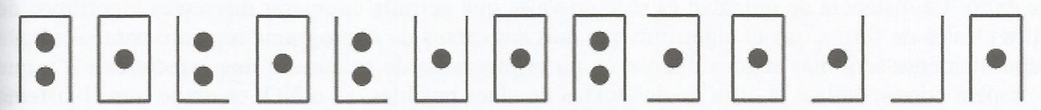
Ejemplo: Cifre con el alfabeto mixto anterior el siguiente mensaje:

M = EN EL ESCARABAJO DE ORO APARECEN SIGNOS DISTINTOS A LOS DEL TEXTO EN CLARO

Solución: Siguiendo el alfabeto de cifra indicado se obtiene el criptograma:

C = }%}#} <»[>[«[♠({ }(>([ )[>}» }%<♣♥ %(<{♣ <0♣%0 (<[#(<{}#0}40(} %{#[> (

Desde el punto de vista histórico, uno de los casos más interesantes de este tipo de cifradores con alfabetos distintos se encuentra en los grabados realizados en 1794 en una lápida del cementerio de *Trinity*, un distrito de Nueva York. El mensaje consistía en un conjunto de símbolos como se indica a continuación.



Sólo 100 años después, en 1896, se logra descifrar este enigma aplicando nociones básicas de estadística y redundancia del lenguaje. Esto indica que, por lo menos para aquella época, el criptosistema empleado aunque hoy en día muy inocente, cumplió con creces el principio de la caducidad de la información. La resolución del criptograma, de gran dificultad debido al pequeño tamaño del mismo, sigue la siguiente clave de asignación de figuras geométricas a los caracteres:

A	B	C
D	E	F
G	H	I/J

K	L	M
N	O	P
Q	R	S

T	U	V
W	X	Y
Z		

Siguiendo entonces esta clave se llega a que el mensaje del criptograma en cuestión es M = REMEMBER DEATH. La dificultad para llegar a este resultado es obvia pues se cuenta con un criptograma demasiado pequeño. En su contra está el hecho de que, precisamente la letra E, la más significativa del inglés, aparece cuatro veces y que el texto que se ha cifrado tiene mucho que ver con el epitafio que alguna mente inquieta pondría en su tumba.

Un sistema muy similar al criptograma de *Trinity*, denominado *Freemasons*, es utilizado por las logias de la masonería. El lector interesado en este tipo de cifrados históricos puede consultar el libro *Cryptology: Machines, History & Methods* reflejado en la bibliografía.

2.4.1.3. Criptoanálisis del cifrado del César

Se comentaba en el apartado anterior que la distancia de unicidad del cifrador del *César* con clave era mayor que en el caso del algoritmo sin clave y, por consiguiente, también su seguridad. No obstante, al producirse en ambos casos una sustitución fija de cada carácter del alfabeto en claro por un único carácter del alfabeto de cifrado, el criptograma podrá romperse fácilmente aplicando técnicas de estadística del lenguaje, siempre y cuando se cuente con una cantidad suficiente de texto cifrado. Como se puede comprobar en el "Apéndice B. Teoría de la información", la distancia de unicidad viene dada por la relación entre la entropía de la clave H(K) y la redundancia del lenguaje D. El siguiente ejemplo muestra este valor para cifradores del tipo *César*.

Ejemplo: Encuentre la distancia de unicidad de un cifrador del *César*.

Solución: Si n = 27, existirán sólo 26 posibles combinaciones de alfabetos, por lo tanto H(K) = $\log_2 26$ = 4,70. Como la redundancia D era igual a 3,4 entonces se tiene que N = H(K)/D ≈ 4,70/3,4 ≈ 1,38. Por lo tanto, se necesitan como mínimo 2 caracteres.

Obviamente con sólo dos caracteres no se puede atacar ningún criptograma con ciertas expectativas de éxito. La distancia de unicidad es sólo un valor que permite comparar diferentes algoritmos de cifra clásica de forma que el algoritmo que más caracteres de criptograma requiere para su ataque supuestamente será más seguro. Piense en un criptograma de solamente dos caracteres FZ y que se espera corresponda a la cifra de dos textos en claro posibles, SI o NO: es obvio que el sistema no tiene solución. Algo parecido sucedía con el criptograma del cementerio de *Trinity* del apartado anterior, era muy difícil criptoanalizarlo pues la cantidad de texto era demasiado pequeña. La pista más valiosa que permitió romperlo fueron esos cuatro símbolos repetidos en una frase de tan solo 13 símbolos.

Para cifrados del *César* sin clave, una forma elemental de criptoanálisis consiste en escribir bajo el texto cifrado todas las combinaciones de frases, con o sin sentido, que se obtienen al aplicar a dicho criptograma desplazamientos de 1, 2, 3, ..., n-1 caracteres, siendo n el número de caracteres del alfabeto utilizado. Lógicamente una de estas combinaciones dará con el texto en claro y esto será válido independientemente del valor asignado a la constante de desplazamiento. Se partirá del siguiente par mensaje/criptograma:

M = CESAR EL EMPERADOR HA SIDO ASESINADO

C = FHVDU HÑHOS HUDGR UKDVL GRDVH VLPDG R

Es posible realizar un criptoanálisis del cifrador del *César* obteniendo el cuadro de todos los posibles mensajes a partir del criptograma. De todas las opciones, la única solución con sentido corresponde a un desplazamiento de 24 caracteres en la operación de descifrado. Si al cifrar el mensaje M se ha desplazado 3 espacios hacia delante para obtener el criptograma C, representado en la posición b = 0, para descifrarlo habrá que desplazarse 3 caracteres hacia atrás o bien, de acuerdo con los principios de la modularidad, un desplazamiento hacia delante de 27-3 = 24 espacios.

b = 0 texto cifrado, b = 24 texto en claro

b = 00	FHVDU	HÑHOS	HUDGR	UKDVL	GRDVH	VLPDG	R
b = 01	GIWEV	IOIPT	IVEHS	VLEWM	HSEWI	WMQEH	S
b = 02	HJXFW	JPJQU	JWFIT	WMFXN	ITFXJ	XNRFI	T
b = 03	IKYGX	KQKRV	KXGJU	XNGYÑ	JUGYK	YÑSGJ	U
b = 04	JLZHY	LRLSW	LYHKV	YÑHZO	KVHZL	ZOTHK	V
b = 05	KMAIZ	MSMTX	MZILW	ZOIAP	LWIAM	APUIL	W
b = 06	LNBJA	NTNUY	NAJMX	APJBQ	MXJBN	BQVJM	X
b = 07	MÑCKB	ÑUÑVZ	ÑBKNY	BQKCR	NYKCÑ	CRWKN	Y
b = 08	NODLC	OVOWA	OCLÑZ	CRLDS	ÑZLDO	DSXLÑ	Z
b = 09	ÑPEMD	PWPXB	PDMOA	DSMET	OAMEP	ETYMO	A
b = 10	OQFNE	QXQYC	QENPB	ETNFU	PBNFQ	FUZNP	B
b = 11	PRGÑF	RYRZD	RFÑQC	FUÑGV	QCÑGR	GVAÑQ	C
b = 12	QSHOG	SZSAE	SGORD	GVOHW	RDOHS	HWBOR	D
b = 13	RTIPH	TATBF	THPSE	HWPIX	SEPIT	IXCPS	E
b = 14	SUJQI	UBUCG	UIQTF	IXQJY	TFQJU	JYDQT	F
b = 15	TVKRJ	VCVDH	VJRUG	JYRKZ	UGRKV	KZERU	G
b = 16	UWLSK	WDWEI	WKSVH	KZSLA	VHSLW	LAFSV	H
b = 17	VXMTL	XEXFJ	XLTWI	LATMB	WITMX	MBGTW	I
b = 18	WYNUM	YFYGK	YMUXJ	MBUNC	XJUNY	NCHUX	J
b = 19	XZÑVN	ZGZHL	ZNVYK	NCVÑD	YKVÑZ	ÑDIVY	K
b = 20	YAOWÑ	AHAIM	AÑWZL	ÑDWOE	ZLWOA	OEJWZ	L
b = 21	ZBPXO	BIBJN	BOXAM	OEXPF	AMXPB	PFKXA	M
b = 22	ACQYP	CJCKÑ	CPYBN	PFYQG	BNYQC	QGLYB	N
b = 23	BDRZQ	DKDLO	DQZCÑ	QGZRH	CÑZRD	RHMZC	Ñ
b = 24	**CESAR**	**ELEMP**	**ERADO**	**RHASI**	**DOASE**	**SINAD**	**O**
b = 25	DFTBS	FMFNQ	FSBEP	SIBTJ	EPBTF	TJÑBE	P
b = 26	EGUCT	GNGÑR	GTCFQ	TJCUK	FQCUG	UKOCF	Q

Para los cifradores por desplazamiento puro como el del *César*, se cumplirá por tanto la siguiente operación de descifrado (D) a partir de un cifrado (E) en el cuerpo n:

$$D_b = E_{n-b} \Rightarrow D_b = E_{27-b}$$

De lo visto anteriormente, es fácil deducir que un sistema de cifra por sustitución monoalfabética como el del *César* presenta un nivel de seguridad mínimo puesto que para romperlo bastará con un lápiz, papel y un poco de paciencia para confeccionar el cuadro anterior, nada del otro mundo como puede ver. Esta debilidad se debe a que el número de desplazamientos posibles es muy pequeño al contar sólo con los 26 valores que corresponden a los caracteres del alfabeto; esto es, se cumple que $1 \leq b \leq 26$, pues un desplazamiento igual a cero o bien múltiplo de veintisiete sería igual que transmitir en claro.

En el caso en que el cifrador del *César* tenga una clave, el ataque anterior no proporciona ninguna solución porque el desplazamiento deja de ser constante y, por lo tanto, es difícil establecer una relación matemática única y directa entre el alfabeto en claro y el alfabeto de cifrado. El único camino que nos queda consiste en llevar las estadísticas del lenguaje al criptograma, observando por ejemplo la frecuencia relativa de aparición de los caracteres en el texto cifrado. Al contrario del método anterior, válido solamente para desplazamientos puros sin clave, este tipo de ataque estadístico será válido tanto para los cifrados de tipo monoalfabético con clave como para aquellos que no la tienen. Ahora bien, en la gran mayoría de los casos será necesario contar con una cantidad de criptograma bastante mayor que la del ejemplo anterior y, cómo no, una pizca de intuición y un poco de suerte.

Ejemplo: Calcule la distancia de unicidad de un cifrador del *César* con clave.

Solución: Si el alfabeto tiene n caracteres, existirán n! combinaciones posibles de n elementos, luego $N = H(K)/D = (\log_2 n!)/D$. Utilizando la aproximación de Sterling, $\log_2 n! \approx n\log_2(n/e)$, la distancia de unicidad será $N \approx n\log_2(n/e)/D$. Para n = 27 se tiene: $N \approx 27\log_2(27/e)/3{,}4 \approx 27{,}4$ caracteres.

Para el cifrador del *César*, al establecerse en la operación de cifrado una correspondencia directa entre los caracteres del texto en claro y del alfabeto de cifrado, se mantiene la misma relación de frecuencia relativa característica del lenguaje. Por lo tanto, es muy probable que la letra C_i del texto cifrado con mayor frecuencia relativa se corresponda con la letra M_i de mayor frecuencia relativa del lenguaje. Esto es, si la letra *W* es la de mayor frecuencia en el criptograma, se puede suponer con muy buenas expectativas de éxito que sea la letra *E* del texto en claro y que, por lo tanto, el desplazamiento aplicado haya sido igual a 19, la distancia que separa ambas letras en nuestro alfabeto. Como ya se ha comentado, estas suposiciones sólo tendrán cierta validez si la cantidad de texto cifrado es grande y por tanto se cumplen las propiedades estadísticas del lenguaje. En el fondo se está realizando una comparación de la distribución de frecuencias de todos los elementos del criptograma con la característica del lenguaje, con el objeto de encontrar ese desplazamiento constante.

Otra forma de atacar un cifrado por desplazamiento puro con o sin clave es buscar digramas, trigramas, ngramas o poligramas y en general palabras características del lenguaje, para asociar un conjunto de caracteres del criptograma con otro conjunto de caracteres del texto en claro. Obviamente este criptoanálisis es también válido para atacar cifrados con alfabetos mixtos como se observa en el siguiente ejemplo.

Ejemplo: Descifre el siguiente criptograma cifrado con un alfabeto mixto en bloque de 5 elementos:

C = ♦>]♣0 ♥34♦3 44]>} >{:☺♠ {4♥3> 34]02 014:0 34(♠4 ♥♣>♠)0]{)4 034014}034 ♦344] 0♥1>]

Solución: Se encuentran las siguientes frecuencias relativas en los caracteres:

4 = 14	0 = 9	3 = 8	> = 6	] = 6
♥ = 4	{ = 3	♠ = 3	♦ = 3	1 = 3
♣ = 2	) = 2	: = 2	} = 2	(= 1
2 = 1	☺ = 1			

Total : 70 caracteres.

Suponiendo que los caracteres 4 y 0, de mayor frecuencia relativa en el criptograma, se corresponden con las letras E y A del alfabeto, se obtiene el siguiente criptograma parcial:
M_1 = ♦>]♣A ♥3E♦3 EE]>} >{:☺♠ {E♥3> 3E]A2 A1E:A
3E(♠E ♥♣>♠) A]{)E A3EA1 E}A3E ♦3EE] A♥1>]

Si por alguna pista se sospecha que]A2A1E:A sea LACABEZA, se tienen otros 4 caracteres y el criptograma sería en una segunda aproximación:
M_2 = ♦>L♣A ♥3E♦3 EE]>} >{:☺♠ {E♥3> 3ELAC ABEZA
3E(♠E ♥♣>♠) AL{)E A3EAB E}A3E ♦3EEL A♥B>L

Ya se han encontrado 6 caracteres con posible equivalencia:
0 = A; 1 = B; 2 = C; 4 = E;] = L y : = Z

De M_2 se podría inferir que 3ELAC ABEZA = DELAC ABEZA es decir que 3 = D. Si además se sospecha que la cadena ELA♥B>L es ELARBOL, el resultado son tres correspondencias nuevas: 3 = D; ♥ = R y > = O. Se obtiene el alfabeto de cifrado que se indica y una tercera aproximación del criptograma:

M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	0	1	2	3	4	_	_	_	_	_	_	]	_	_	_	>	_	_	♥	_	_	_	_	_	_	_	:

M_3 = ♦OL♣A RDE♦D EEL>} >{Z☺♠ {ERDO DELAC ABEZA
DE(♠E R♣O♠) AL{)E ADEAB E}ADE ♦DEEL ARBOL

Si ABE}ADE♦DEELARBOL es ABEJADESDEELARBOL, se obtienen dos nuevas correspondencias entre caracteres } = J y ♦ = S y el siguiente criptograma:
M_4 = SOL♣A RDESD EEL>J >{Z☺♠ {ERDO DELAC ABEZA
DE(♠E R♣O♠) AL{)E ADEAB EJADE SDEEL ARBOL

La resolución final del criptograma a estas alturas parece ya algo trivial. El alfabeto utilizado en este ejemplo y el mensaje son los que se indican:

M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	0	1	2	3	4	5	«	»	{	}	[	]	(	)	<	>	J	☺	♥	♦	♣	♠	$	&	@	#	:

M = SOLTAR DESDE EL OJO IZQUIERDO DE LA CABEZA DE MUERTO UNA LÍNEA DE ABEJA DESDE EL ÁRBOL

La técnica de criptoanálisis del ejemplo anterior es precisamente la usada por Allan Poe en su cuento "El escarabajo de oro". El texto corresponde a una parte de ese enigmático mensaje con un alfabeto de cifrado ligeramente distinto.

Como conclusión se puede afirmar que incluso incluyendo una clave en la cifra, por complicada y larga que ésta sea, estos criptosistemas monoalfabéticos son todos muy vulnerables a los ataques de

un criptoanalista. Si además se cuenta con la ayuda de un simple ordenador, el ataque y posterior solución a estos criptogramas se convierte en la práctica en un juego.

2.4.1.4. Cifradores genéricos por sustitución

Ya se ha comentado en el apartado anterior qué se entiende por un cifrador monoalfabético por sustitución, por lo menos para el caso del cifrador del *César* en el que la sustitución de los caracteres se obtiene por medio de un desplazamiento constante en el alfabeto. A continuación se analizarán los cifradores monoalfabéticos genéricos, también conocidos como cifradores de transformaciones afines. En este caso la operación de sustitución de los caracteres del alfabeto puede obtenerse de forma matemática aplicando la siguiente expresión de equivalencia:

$$C_i = (a*M_i + b) \bmod n$$

en donde a se conoce como la constante de decimación, b constante de desplazamiento y n es el cuerpo de cifra. Observe que de acuerdo con la ecuación anterior la relación matemática del cifrador del *César* será:

$$\text{Cifrador del } \textit{César}\text{:}\quad C_i = (M_i + 3) \bmod n$$

¿Puede utilizarse cualquier valor de a y b en la ecuación anterior? La respuesta es no. En primer lugar, es obvio que a no puede ser igual a cero pues no existiría una equivalencia de alfabetos por lo que deberá cumplirse que $a \geq 1$. Por otra parte, para la existencia de inversos deberá cumplirse que los valores de la constante a y el módulo n sean primos entre sí; es decir el máximo común divisor sea 1, mcd (a, n) = 1. Al trabajar en módulo $27 = 3^3$, los valores permitidos de la constante de decimación a serán los 18 elementos del Conjunto Reducido de Restos CRR (27) que no tengan como factor común el número 3, es decir: 1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 20, 22, 23, 25 y 26.

En cuanto a la constante de desplazamiento b, ésta puede tomar cualquier valor comprendido entre 0 y n-1 pues se asegura en todo momento la existencia del inverso para la adición. Desplazamientos mayores que n-1 caerán dentro de la misma clase de equivalencia por lo que su valor se reduce módulo n. Por ejemplo, un desplazamiento de b = 32 espacios equivale a (32-k*n) = (32-1*27) = 5 espacios efectivos. Así mismo, un desplazamiento negativo (caracteres hacia la izquierda del alfabeto) puede trasladarse a su equivalente en el intervalo [0, n-1]. Por lo tanto, se puede decir que un desplazamiento de b = -8 caracteres equivale a (-8+k*n) = (-8+1*27) = 19 caracteres hacia la derecha.
De esta manera, cuando la constante de decimación a es igual a la unidad, el cifrador genérico de sustitución se transforma en uno de desplazamiento puro; cuando la constante de desplazamiento b es igual a cero se trata de cifradores por decimación pura y cuando se cumple que la constante a es mayor que la unidad y b es distinto de cero, la cifra es por sustitución afín.

Cifradores por desplazamiento puro

Corresponden a los denominados cifradores tipo *César* ya vistos en el apartado anterior por lo que no se va a repetir lo allí comentado. Las operaciones de cifra y descifrado serán:

$$Ci = (Mi + b) \bmod n$$

$$M_i = (C_i - b) \bmod n$$
$$M_i = (C_i + n - b) \bmod n$$

Ejemplo: a) Con n = 27 y un desplazamiento b = 15, cifre y luego descifre el mensaje M = SALVE CÉSAR.

Solución: a) $C_i = (m_i + 15) \bmod 27$

C_{01} = (S+15) mod 27 = (19+15) mod 27 = 34 mod 27 = 07 = H
C_{02} = (A+15) mod 27 = (00+15) mod 27 = 15 mod 27 = 15 = O
C_{03} = (L+15) mod 27 = (11+15) mod 27 = 26 mod 27 = 26 = X
C_{04} = (V+15) mod 27 = (21+15) mod 27 = 36 mod 27 = 09 = J
C_{05} = (E+15) mod 27 = (04+15) mod 27 = 19 mod 27 = 19 = S
C_{06} = (C+15) mod 27 = (02+15) mod 27 = 17 mod 27 = 17 = Q
C_{07} = (E+15) mod 27 = (04+15) mod 27 = 19 mod 27 = 19 = S
C_{08} = (S+15) mod 27 = (19+15) mod 27 = 34 mod 27 = 07 = H
C_{09} = (A+15) mod 27 = (00+15) mod 27 = 15 mod 27 = 15 = O
C_{10} = (R+15) mod 27 = (18+15) mod 27 = 33 mod 27 = 06 = G

El criptograma será: C = HOXJS QSHOG.

b) $M_i = (c_i + 27 - 15) \bmod 27 = (c_i + 12) \bmod 27$

M_{01} = (H+12) mod 27 = (07+12) mod 27 = 19 mod 27 = 19 = S
M_{02} = (O+12) mod 27 = (15+12) mod 27 = 27 mod 27 = 00 = A
M_{03} = (X+12) mod 27 = (26+12) mod 27 = 38 mod 27 = 11 = L
M_{04} = (J+12) mod 27 = (09+12) mod 27 = 21 mod 27 = 21 = V
M_{05} = (S+12) mod 27 = (19+12) mod 27 = 31 mod 27 = 04 = E
M_{06} = (Q+12) mod 27 = (17+12) mod 27 = 29 mod 27 = 02 = C
M_{07} = (S+12) mod 27 = (19+12) mod 27 = 31 mod 27 = 04 = E
M_{08} = (H+12) mod 27 = (07+12) mod 27 = 19 mod 27 = 19 = S
M_{09} = (O+12) mod 27 = (15+12) mod 27 = 27 mod 27 = 00 = A
M_{10} = (G+12) mod 27 = (06+12) mod 27 = 18 mod 27 = 18 = R

El mensaje descifrado es: M = SALVE CÉSAR.

Cifradores por decimación pura

Ya se ha visto que si la constante de decimación es igual a la unidad, el cifrador se transforma en uno de desplazamiento puro como el del *César*. Si además la constante de desplazamiento es cero, entonces se transmite en claro lo que no tiene sentido criptográfico. Por el contrario, si la constante de decimación es

mayor que la unidad y la constante de desplazamiento es cero, se tratará de un cifrador por decimación pura. En este caso las ecuaciones de cifra y descifrado serán:

$$C_i = a * M_i \text{ mod } n$$
$$M_i = a^{-1} * C_i \text{ mod } n$$

donde a^{-1} será el inverso del factor de decimación en el cuerpo n; es decir inv (a, n).

Ejemplo: a) Encuentre el alfabeto de cifrado para la transformación monoalfabética por decimación $C_i = 20 * M_i$ mod 27.

b) Cifre con este alfabeto el siguiente mensaje M = DILE A LAURA QUE LA QUIERO

Solución: a) El alfabeto de cifra será:

C_0 = 20*A mod 27 = 20*0 mod 27 = 0 = A

C_1 = 20*B mod 27 = 20*1 mod 27 = 20 = T

C_2 = 20*C mod 27 = 20*2 mod 27 = 40 mod 27 = 13 = N, etc.

–	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	A	T	N	G	Z	S	M	F	Y	R	L	E	X	Q	K	D	W	P	J	C	V	O	I	B	U	Ñ	H

b) El criptograma será C = GYEZA EAOJA POZEA POYZJ D.

Puesto que sólo se aplica este algoritmo de cifra si el valor de la constante de decimación es primo relativo con el módulo de trabajo, se asegura la existencia del inverso y por tanto la posibilidad de descifrar el criptograma.

Ejemplo: Descifre el criptograma que se indica si se conoce que éste se ha obtenido con sustitución por decimación pura con una constante de decimación igual a 22.

C = MÑZHW DHBGR HZZAU DHRG.

Solución: El inverso del factor de decimación 22 es inv (22, 27) = 16, como M=12, Ñ=14, Z=26, etc., se tiene:

M_{01} = 16 * 12 mod 27 = 192 mod 27 = 03 = D

M_{02} = 16 * 14 mod 27 = 224 mod 27 = 08 = I

M_{03} = 16 * 26 mod 27 = 416 mod 27 = 11 = L

M_{04} = 16 * 07 mod 27 = 112 mod 27 = 04 = E

M_{05} = 16 * 23 mod 27 = 368 mod 27 = 17 = Q

M_{06} = 16 * 03 mod 27 = 048 mod 27 = 21 = U

M_{07} = 16 * 07 mod 27 = 112 mod 27 = 04 = E

M_{08} = 16 * 01 mod 27 = 016 mod 27 = 16 = P

M_{09} = 16 * 06 mod 27 = 096 mod 27 = 15 = O

M_{10} = 16 ∗ 18 mod 27 = 288 mod 27 = 18 = R
M_{11} = 16 ∗ 07 mod 27 = 112 mod 27 = 04 = E
M_{12} = 16 ∗ 26 mod 27 = 416 mod 27 = 11 = L
M_{13} = 16 ∗ 26 mod 27 = 416 mod 27 = 11 = L
M_{14} = 16 ∗ 00 mod 27 = 000 mod 27 = 00 = A
M_{15} = 16 ∗ 21 mod 27 = 336 mod 27 = 12 = M
M_{16} = 16 ∗ 03 mod 27 = 048 mod 27 = 21 = U
M_{17} = 16 ∗ 07 mod 27 = 112 mod 27 = 04 = E
M_{18} = 16 ∗ 18 mod 27 = 288 mod 27 = 18 = R
M_{19} = 16 ∗ 06 mod 27 = 096 mod 27 = 15 = O

El mensaje descifrado es M = DILE QUE POR ELLA MUERO

Cifradores por transformación afín

En los cifradores genéricos, si se cumple que la constante de decimación *a* es mayor que 1 y la constante de desplazamiento *b* distinto de cero, se habla de cifra por transformación afín. Las ecuaciones serán en este caso:

$$C_i = (a*M_i + b) \bmod n$$
$$M_i = a^{-1} (C_i - b) \bmod n$$

La ecuación de descifrado también puede escribirse como sigue:

$$M_i = a^{-1} (C_i + n - b) \bmod n$$

Ejemplo: Encuentre el alfabeto de cifrado monoalfabético para la siguiente relación de transformación

$$C_i = (4*M_i + 5) \bmod 27.$$

Solución:

C_0 = (4∗A + 5) mod 27 = (4∗0 + 5) mod 27 = 5 = F
C_1 = (4∗B + 5) mod 27 = (4∗1 + 5) mod 27 = 9 = J
C_2 = (4∗C + 5) mod 27 = (4∗2 + 5) mod 27 = 13 = N, etcétera.

-	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	F	J	N	Q	U	Y	C	G	K	Ñ	R	V	Z	D	H	L	O	S	W	A	E	I	M	P	T	X	B

Del ejemplo anterior, observe que el desplazamiento indica dónde comienza la secuencia del alfabeto de cifrado y, por su parte, la decimación muestra los saltos que va dando un carácter en el alfabeto original para recorrerlo en su totalidad. ¿Qué sucederá si aplica una relación de trasformación monoalfabética que no cumpla con las condiciones anteriores? El siguiente ejemplo aclarará esta situación.

Ejemplo: Encuentre el alfabeto de cifrado monoalfabético para la siguiente relación de transformación:

$$C_i = (3*M_i + 2) \bmod 27.$$

Solución: Aplicando la ecuación se obtiene:

	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	C	F	I	L	Ñ	Q	T	W	Z	C	F	I	L	Ñ	Q	T	W	Z	C	F	I	L	Ñ	Q	T	W	Z

¿Qué sucede en este caso? Con los valores anteriores, no es válida la relación de transformación de alfabetos pues mcd (3,27) ≠ 1 lo que se comprueba en el hecho de que no se obtiene una equivalencia unívoca entre caracteres. Para este caso en que $a = 3$, se va repitiendo el mismo conjunto de 9 (27/3) caracteres *CFILÑQTWZ* por lo que no se obtiene el conjunto completo de restos del módulo y, por tanto, no es posible utilizarlo como cifrador. De hecho si el criptograma presenta la letra C, no se sabe si corresponde a la acción de cifrar en el texto en claro las letras J, R o A.

Ejemplo: Usando un cifrador monoalfabético por decimación y adición según la transformación $(5*M_i + 8) \bmod 27$, cifre el siguiente mensaje: M = DÁBALE ARROZ A LA ZORRA EL ABAD.

Solución:

	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	I	N	R	W	B	G	L	P	U	Z	E	J	Ñ	S	X	C	H	M	Q	V	A	F	K	O	T	Y	D

Luego, C = WINIJ BIQQC DIJID CQQIB JINIW.

Al igual que en el cifrador del *César*, puede incluirse una clave secreta para aumentar la seguridad del sistema. Primero se aplica la relación de transformación para encontrar un alfabeto de cifrado, luego se escribe la clave a partir de una posición p_0 y finalmente se desplazan los caracteres restantes del alfabeto de cifrado encontrado a partir de la posición final de la clave como se muestra en el siguiente ejemplo.

Ejemplo: Aplicando la transformación $C_i = (7*M_i + 2) \bmod 27$ conjuntamente con la clave K = REFRANERO ESPAÑOL posicionada en $p_0 = 5$, se pide cifrar el mensaje M = NO HAY MAL QUE POR BIEN NO VENGA.

Solución: Aplicando la transformación $(7*M_i + 2) \bmod 27$, se obtiene el siguiente alfabeto de cifrado:

M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	C	J	P	W	D	K	Q	X	E	L	R	Y	F	M	S	Z	G	N	T	A	H	Ñ	U	B	I	O	V

A continuación se escribe la clave en $p_0 = 5$:



Completando ahora el alfabeto de cifrado desde la posición final de la clave, se obtiene:

M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	H	U	B	I	V	R	E	F	A	N	O	S	P	Ñ	L	C	J	W	D	K	Q	X	Y	M	Z	G	T

Luego, C = ÑCFHG PHSWX VJCDU AVÑÑC YVÑEH.

La transformación anterior sigue manteniendo una relación monoalfabética, independientemente de cómo se distribuyan los caracteres; es más, el alfabeto de cifrado no tiene por qué seguir una lógica ni mucho menos una relación matemática como parece deducirse por lo visto; basta con que exista una relación de uno a uno entre alfabeto en claro y alfabeto de cifrado. Lo único que se logra con ello es cambiar la relación de correspondencia de caracteres y, en última instancia, ponerle las cosas algo más difíciles a nuestro enemigo criptoanalista, desgraciadamente sólo un poco.

Puesto que estos cifradores monoalfabéticos genéricos no se diferencian mucho de los vistos en los apartados anteriores, tendrán la misma fragilidad ante un hipotético ataque de un criptoanalista.

2.4.1.5. Criptoanálisis de cifrados monoalfabéticos por sustitución

El criptoanálisis de los cifradores monoalfabéticos genéricos por sustitución, esto es aquellos en los que se cumple la relación afín (a*M + b) mod n, pueden resolverse fácilmente aplicando estadísticas del lenguaje al igual que en el cifrado del *César*. En este caso, como además de un desplazamiento b existe una constante de decimación a, es posible plantear un sistema de dos ecuaciones para asignar posibles valores a ambas variables en función del comportamiento estadístico que se observa en los caracteres del criptograma. Como la transformación es $C_i = (a*M_i + b) \bmod n$, se asocia, según la frecuencia relativa de aparición de caracteres en el criptograma, valores de posición de dichos caracteres con los del alfabeto en claro. A continuación se hace un ensayo con diferentes relaciones de congruencia, a partir de unas supuestas correspondencias entre caracteres del criptograma con los caracteres del texto en claro, para ver cómo funciona este método de ataque.

Si se sospecha que la letra más frecuente en un criptograma cualquiera, por ejemplo la letra M = 12, se corresponde con la letra más frecuente en el lenguaje castellano, es decir la letra E = 4, hay la primera relación de equivalencia:

$$a*4 + b = 12 \bmod 27$$

Siguiendo con la característica de los monogramas en castellano, si se cree ahora que existe una relación directa entre el segundo carácter más frecuente del criptograma, por ejemplo la letra G = 6, con la segunda letra más frecuente del lenguaje, A = 0. Esto nos da una segunda ecuación:

$$a*0 + b = 6 \bmod 27$$

Se deduce que b = 6. Si se reemplaza este valor en la primera ecuación:

$$a*4 + 6 = 12 \bmod 27 \quad \Rightarrow \quad a*4 = 6 \bmod 27$$

$$a = 6*\mathrm{inv}(4,27) \bmod 27 \quad \Rightarrow \quad a = 6*7 \bmod 27 \Rightarrow a = 15$$

Este resultado no nos sirve pues si a = 15 entonces mcd (a, n) = 3 y esto no puede dar lugar a un alfabeto de cifrado pues no genera el CCR (27). A un resultado similar se habría llegado si, por ejemplo, manteniendo la relación de la primera ecuación se hubiese supuesto una correspondencia entre la letra R = 18 del criptograma y la letra C = 2 del texto en claro.

Para no aburrirle con esto, suponga ahora que las correspondencias válidas observadas entre caracteres del criptograma y texto en claro son las siguientes:

M = 12 del criptograma se corresponde con E = 4 del texto en claro.

T = 20 del criptograma se corresponde con R = 18 del texto en claro.

Se establece así el siguiente sistema de ecuaciones:

$$a*4 + b = 12 \bmod 27$$
$$a*18 + b = 20 \bmod 27$$

Restando la primera de la segunda se tiene:

$$a*14 = 8 \bmod 27 \Rightarrow a = 8*[\text{inv}(14,27)] \bmod 27 = 8*2 \bmod 27 = 16$$

Sustituyendo a = 16 en una de las ecuaciones, se obtiene b = 2. Como ahora mcd (16, 27) = 1, entonces el sistema de ecuaciones encontrado, sí podría entregar una solución válida. En este caso la transformación de cifrado aplicada podría ser:

$$C_i = (16*M_i + 2) \bmod 27$$

Si esta congruencia no nos entrega un texto en claro con sentido, habrá que buscar otras correspondencias entre caracteres del criptograma y del alfabeto, según su distribución de frecuencias, y volver a plantear el sistema de ecuaciones hasta encontrar la transformación que dé lugar al mensaje esperado. Observe que, aunque las relaciones de congruencia sean válidas, no por ello dicha transformación dará lugar a una solución válida en el espacio de los mensajes.

Ejemplo: Descifre el siguiente criptograma obtenido mediante una transformación monoalfabética por decimación y desplazamiento sin clave: C = NAQÑF EKNDP NCIVU FPUAN EJUIP FCNER NFRÑF UNPLN AFPFQ TFPEI JRTÑE FPKÑI KTAPF LIKIÑ AIPÑU RCUJI PCIVU CUNER IRLNP TJIAF NEOIÑ CFLNC NLUFA TEF.

Solución: Los caracteres de mayor frecuencia del criptograma anterior son: F = 14, N = 13 e I = 12. Esto nos hace sospechar que se correspondan con las letras A, E y O del alfabeto en claro. Se harán sólo dos intentos para mostrar cómo funciona este método de ataque:

1ª Aproximación (que nos lleva a una solución falsa):

Texto claro: E(4) ⇒ Criptograma: F(5)

Texto claro: A(0) ⇒ Criptograma: N(13)

Texto claro: O(15) ⇒ Criptograma: I(8)

Luego:

$(a*4 + b) = 5 \bmod 27$

$(a*0 + b) = 13 \bmod 27 \Rightarrow b = 13$

$(a*15 + b) = 8 \bmod 27$

Restando la primera ecuación a la tercera: $a*11 = 3 \bmod 27$.

Luego, $a = (3*\text{inv}(11,27)) \bmod 27 = 3*5 \bmod 27 = 15 \Rightarrow a = 15$.

La solución $E(M) = (15M_i + 13)$ se descarta pues mcd $(15,27) \neq 1$.

2ª Aproximación (que nos lleva ahora sí a una solución verdadera):

Texto claro: E(4) $\Rightarrow$ Criptograma: N(13)
Texto claro: A(0) $\Rightarrow$ Criptograma: F(5)
Texto claro: O(15) $\Rightarrow$ Criptograma: I(8)

Luego:
$(a*4 + b) = 13 \bmod 27$
$(a*0 + b) = 5 \bmod 27 \Rightarrow b = 5$
$(a*15 + b) = 8 \bmod 27$

Restando la primera ecuación a la tercera: $a*11 = 22 \bmod 27$:
Luego $a = (22*inv(11,27)) \bmod 27 = 22*5 \bmod 27 \Rightarrow a = 2$.

Esto da lugar a $E(M) = (2M_i + 5) \bmod 27$, cuyo alfabeto es:

M_i	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_i	F	H	J	L	N	O	Q	S	U	W	Y	A	C	E	G	I	K	M	Ñ	P	R	T	V	X	Z	B	D

Luego el criptograma se descifra como:
M = EL GRAN PEZ SE MOVÍA SILENCIOSAMENTE A TRAVÉS DE LAS AGUAS NOCTURNAS, PROPULSADO POR LOS RÍTMICOS MOVIMIENTOS DE SU COLA EN FORMA DE MEDIA LUNA. (Primer párrafo del libro "Tiburón", de *P. Benchley*).

En el caso en que se utilice una clave, el método anterior falla porque deja de existir una relación matemática directa y constante entre el alfabeto en claro y el alfabeto de cifrado. La única solución de ataque, siempre y cuando se conozca o sospeche que se trata de un cifrado monoalfabético, consistirá en buscar digramas, trigramas, y en general formación de palabras características del lenguaje repetidos en el criptograma, evidentemente con caracteres diferentes, y de esta forma obtener el alfabeto de cifrado como ya ha sido resuelto en un ejemplo de apartado 2.4.1.3. En este escenario la tarea será más tediosa que la anterior pero, no obstante, relativamente sencilla para este tipo de cifradores.

¿Qué sucede con la distancia de unicidad en estos criptosistemas genéricos? Si se analizan las distintas posibilidades de alfabetos de cifrado, se observa que las posibles transformaciones están ligadas directamente con el factor de decimación. Esto es, la constante de desplazamiento puede tener cualquier valor puesto que asegura una operación inversa; sin embargo, la constante de decimación debe cumplir la condición de ser primo relativo con el módulo. Esto va a indicar que las combinaciones de alfabetos de cifrado serán $n*\phi(n)$; es decir, los n posibles desplazamientos por el indicador de *Euler* que indica el número de elementos que contiene el CRR (n), es decir primos con el módulo y que por tanto sirven como factor de decimación.

Ejemplo: Encuentre la distancia de unicidad de un cifrador genérico de sustitución sin clave para n = 27.

Solución: $N = H(K)/D = \log_2(n\phi(n))/D = \log_2(27*18)/3{,}4 = 2{,}6$. Esto es, es necesario contar con 3 caracteres cifrados como mínimo.

El valor de la distancia de unicidad para estos cifradores afines es muy bajo puesto que también lo es la cantidad de alfabetos. Con un valor de n = 27, solamente existen 486 alfabetos distintos, es decir, 26 desplazamientos por 18 decimaciones. Para hacer crecer este valor se puede usar una clave. En esta situación, el cifrador se convierte en uno monoalfabético con clave, con $27! \approx 10^{28}$ alfabetos diferentes, un valor considerable no cabe duda y que nos entrega una distancia de unicidad igual a 28 caracteres.

Como se comprueba, cualquier cifrador por sustitución monoalfabeto, bien sea por decimación, por desplazamiento o genérico con ambas transformaciones, incluso cuando se utiliza una clave, es muy poco seguro y su ataque en muchos casos se convierte en un paseo para el criptoanalista. El principal problema de estos cifradores está en que el texto cifrado es un fiel reflejo del lenguaje, manifestándose las características de redundancia del lenguaje. La primera solución que se nos puede ocurrir, que no la óptima por cierto, para evitar que en el criptograma se vea reflejada la redundancia del lenguaje es la utilización de homófonos, lo que da lugar a este tipo de cifradores que se verán a continuación.

2.4.2 Cifradores por homófonos

2.4.2.1 Cifradores por homófonos de primer orden

¿Qué se entiende por homófonos? La definición puede encontrarse en cualquier diccionario: "palabras de igual pronunciación o sonido y distinto significado" como por ejemplo hola-ola, barón-varón, rallar-rayar, etc. En criptografía se entiende por homófonos a las distintas representaciones que se dan al mismo carácter sin seguir ninguna relación o función determinada. Por ejemplo, si se establece una relación entre los 27 caracteres del alfabeto con los 100 primeros números del 00 al 99, se podría asociar a la letra A los siguientes números: 3, 16, 19, 24, 56, 71, 73, 88, 97. Luego, el receptor autorizado al conocer esta correspondencia simplemente reemplaza dichos números por la letra A para descifrar el mensaje. Esto da lugar a los denominados cifradores por homófonos, cuya característica principal es la de suavizar la distribución de frecuencias típica del lenguaje, de forma que el criptoanalista no pueda emplear las técnicas estadísticas vistas en los apartados anteriores. La técnica consiste entonces en asignar un mayor número de homófonos a los caracteres más frecuentes del lenguaje y un menor número a aquellos menos frecuentes, con el objeto de que la distribución de estos valores se asemeje lo más posible a una distribución normal.

Observe que en tanto un carácter del texto en claro se cifrará con más de un carácter o símbolo del alfabeto de cifrado, no se estaría ya en presencia de un criptosistema monoalfabético. Además, el algoritmo de cifra no tiene porqué seguir una función determinada de asignación de homófonos durante el proceso. Por lo tanto, en este tipo de cifrados se hace corresponder cada uno de los caracteres del alfabeto del texto en claro con un conjunto de elementos f(a) denominados homófonos y que pueden ser cualquier tipo de signos, figuras o números. Luego, si el mensaje M está compuesto por los elementos $\{M_1, M_2, ..., M_m\}$, entonces se tiene que $\{C_1, C_2, ..., C_m\}$ será el conjunto de elementos del criptograma en donde cada C_i se toma a partir de un conjunto de homófonos para $f(M_i)$. Esto quiere decir que un mensaje M con una cadena de caracteres $M_1M_2M_3...M_m$ se cifrará como se indica:

$$M = M_1M_2M_3...M_m \quad \Rightarrow \quad C = f(M_1)\, f(M_2)\, f(M_3)...\, f(M_m)$$

Por ejemplo, si se asignan los homófonos que se adjuntan, se podría cifrar el mensaje M = BÁJAME LA JAULA JAIME como se indica:

A ⇒	03	19	36	83	91			
B ⇒	23	62						
E ⇒	07	25	28	62	70	88	89	97
I ⇒	13	55	70					
J ⇒	43							
L ⇒	18	53	60					
M ⇒	10	33	71	80				
U ⇒	06							

M = B A J A M E L A J A U L A J A I M E
C = 62 36 43 03 71 25 18 91 43 19 06 53 83 43 83 55 10 97

No obstante, cualquier criptograma que respete la asignación antes indicada es también válido, pues el receptor autorizado conocerá dicha tabla de homófonos y podrá descifrar el criptograma. Es decir, para el ejemplo anterior, también es válida entre otras la siguiente transformación:

C = 23 03 43 91 33 07 18 36 43 03 06 60 83 43 36 13 10 70.

Uno de los cifradores homofónicos más conocidos en la historia de la criptografía es el atribuido al aventurero *Thomas Jefferson Beale*, quien en 1821 deja tres mensajes cifrados, llamados respectivamente B_1, B_2 y B_3, en el que supuestamente entrega todas las pistas para descubrir un fabuloso tesoro por él enterrado en Virginia, Estados Unidos. La técnica aplicada por *Beale* para formar el conjunto de homófonos del cifrado B_2 no deja de ser sorprendente: asigna números a los caracteres del alfabeto según la posición de la palabra en cuestión que comienza con dicha letra dentro del texto de la Declaración de la Independencia de los Estados Unidos, cuyas 107 primeras palabras se muestran a continuación:

When, in the course of human events, it becomes necessary	(01-10)
for one people to dissolve the political bands which have	(11-20)
connected them with Another, And to Assume Among the Powers	(21-30)
of the earth the separate And equal station to wich	(31-40)
the Laws of Nature And of Nature's God entitle them,	(41-50)
A decent respect to the opinions on mankind requires that	(51-60)
they should declare the causes wich impel them to the	(61-70)
separation. We hold these truths to be self-evident; that	(71-80)
All men Are created equal, that they Are endowed by	(81-90)
their Creator with certain unalienable rights; that Among these are	(91-100)
Life, Liberty, and the pursuit of Happiness.	(101-107)

Por ejemplo, siguiendo el texto de la Declaración de la Independencia de los Estados Unidos se obtienen los homófonos de valor menor que 100 para las letras A, B, C, D y E que se recogen a continuación, marcadas en mayúsculas y negrita en el texto para homófonos de la letra A:

A 24, 25, 27, 28, 36, 45, 51, 81, 83, 88, 98
B 9, 18, 77, 90
C 4, 21, 65, 84, 92, 94
D 15, 52, 63
E 7, 33, 37, 49, 79, 85, 89

Así, el cifrado B_2 que especificaba el valor del tesoro y cómo había sido enterrado, comenzaba de la siguiente forma: "*I have deposited...*" y terminaba con el siguiente mensaje: "*Paper Number One describes the exact locality of the vault, so that no difficulty will be hand in finding it*" (su traducción es elemental). Esto trajo de cabeza a criptoanalistas aficionados cuya mayor ilusión era encontrar dicho tesoro. Tras diversos estudios más serios por parte del Laboratorio de Criptografía George Fabyan en Riverbank Illinois, se llega a la conclusión de que B_1 sigue el mismo principio de cifrado que B_2 pero, por muchos intentos que se hacen, no se llega a descifrarlo. *James J. Gillogly* y *Louis Kruh*, exponen en 1980 y 1982, respectivamente, en sendos artículos de la revista *Cryptologia*, las anomalías encontradas en el primer criptograma de *Beale*, llegando a la conclusión de que se trata de una gran broma, posiblemente llevada a cabo por *James B. Ward*, vecino de Campbell County en Virginia, a quien supuestamente habían llegado los criptogramas de mano de *Robert Morris*, el tabernero a quien *Beale* habría confiado su secreto al abandonar el pueblo... algo enrevesado, pero que no deja de ser curioso.

El problema de la generación de homófonos a partir de un texto está en que, salvo que éste tenga una extensión muy grande, no se consiguen homófonos para algunas letras pocas frecuentes como inicios de palabras, como sería el caso de las letras K, Ñ y W para el castellano. La única solución consistiría en dejar unos números al final del cuerpo de homófonos para estos caracteres poco frecuentes.

Ejemplo: Construya una tabla de homófonos con las 99 primeras palabras del libro "Cien años de soledad" de *Gabriel García Márquez* y luego cifre el siguiente mensaje: M = UNA GRAN NOVELA.

Solución: El texto indicado es el siguiente:

"Muchos años después, frente al pelotón de fusilamiento, el coronel 10
Aureliano Buendía había de recordar aquella tarde remota en que 20
su padre lo llevó a conocer el hielo. Macondo era 30
entonces una aldea de veinte casas de barro cañabrava construidas 40
a la orilla de un río de aguas diáfanas que 50
se precipitaban por un lecho de piedras pulidas, blancas y 60
enormes como huevos prehistóricos. El mundo era tan reciente, que 70
muchas cosas carecían de nombre, y para mencionarlas había que 80
señalarlas con el dedo. Todos los años, por el mes 90
de marzo, una familia de gitanos desarrapados plantaba su ..." 99

Los caracteres con homófonos serán en este caso:

A	2 5, 11, 16, 25, 33, 41, 48, 87
B	12, 38, 59
C	10, 26, 36, 39, 40, 62, 72, 73, 82
D	3, 7, 14, 34, 37, 44, 47, 49, 56, 74, 84, 91, 95, 97
E	9, 19, 27, 30, 31, 61, 65, 67, 83, 89
F	4, 8, 94
G	96
H	13, 28, 63, 79
L	23, 24, 42, 55, 86
M	1, 29, 66, 71, 78, 90, 92
N	75
O	43
P	6, 22, 52, 53, 57, 58, 64, 77, 88, 98
Q	20, 50, 70, 80
R	15, 18, 46, 69
S	21, 51, 81, 99
T	17, 68, 85
U	32, 45, 54, 93
V	35
Y	60, 76

Uno de los tantos criptogramas podría ser: C = 32 75 11 96 46 33 75 75 43 35 30 55 48.

Como puede apreciar, en el ejemplo anterior no se ha podido encontrar homófonos para todo el alfabeto por lo que no podría cifrar, por ejemplo, el mensaje M = UNA OBRA MAGISTRAL al no tener homófono la letra I. Además de lo anterior, si bien el método utilizado por *Beale* para la generación de homófonos entrega un cifrado que es difícil romper, cumple sólo parcialmente con el principio básico de estos cifradores, cual es la destrucción de la distribución característica de los caracteres a través de una distribución plana de los mismos en el criptograma. Esto es, si en un determinado lenguaje (castellano por ejemplo en módulo 27) las letras P, U, R y A presentan unos valores aproximados de frecuencia relativa iguales a 3, 4, 7 y 11 por ciento, respectivamente, entonces sobre 100 números o signos elegidos como homófonos, se deberían asignarían por ejemplo 3 homófonos a la letra P, 4 a la letra U, 7 a la R y 11 a la letra A. En el método propuesto por *Beale*, no se consigue esta distribución de homófonos.

De los ejemplos anteriores, ninguno de los dos textos tomados como referencia para los homófonos -la Declaración de la Independencia de los Estados Unidos en el primero y el libro de *García Márquez* en el segundo- cumplen con esto. Entre otras diferencias notables, en ambos casos la letra

A está por encima de la letra E, lo que no se corresponde ni con el lenguaje inglés ni el castellano. La única ventaja que tiene elegir estos textos como generadores de homófonos está en la sencillez del algoritmo de asignación y he aquí, precisamente, su gran debilidad puesto que el criptoanalista puede llegar a descubrir toda la clave por intuición a partir de un pequeño trozo por todos conocidos. Por ejemplo, está claro que el texto «En un lugar de La Mancha ...» es una malísima elección puesto que es una pista excelente para un probable criptoanalista. Un sistema de homófonos con una mayor seguridad sería, por ejemplo, asignar números de tres dígitos a las letras del alfabeto, obteniendo dichos números a partir de una página en particular de una Guía de Teléfonos; si es de otro país y antigua mejor aún. Dicha posición o página sería la clave secreta del criptosistema en cuestión. El problema será ahora custodiar adecuadamente ese libro/clave.

2.4.2.2 Cifradores por homófonos de orden mayor

A mayor cantidad de texto cifrado, existe una mayor facilidad para abordar el criptoanálisis. La razón es que una única clave descifra el criptograma C en un texto con sentido en el espacio de los mensajes M. Esto se ve agravado si la clave está asociada con algún documento o texto ampliamente conocido como ya se ha comentado. El método de cifrado con homófonos de mayor orden intenta solucionar este problema. La idea es que un mismo criptograma pueda ser descifrado con claves diferentes y produzca en cada caso un mensaje con sentido en el espacio M con igual probabilidad.

Cifrador homofónico de segundo orden

En este cifrador se realiza la asignación de homófonos mediante una cuadrícula de forma que dicho valor representa a una letra si se lee a través de las filas y otra letra distinta si la lectura se hace a través de las columnas. Así, se envía el mensaje verdadero y uno falso, ambos cifrados con el valor de dicha cuadrícula, de forma que el criptoanalista en el mejor de los casos podrá contar con dos mensajes, sin saber cuál de ellos es el verdadero. El algoritmo es el siguiente:

a) Los números 1 al n^2 se distribuyen de forman aleatoria en una matriz K de orden n*n, cuyas filas y columnas corresponden a los caracteres del alfabeto.

b) Para cada carácter *a* del alfabeto, la fila de la matriz K define un conjunto de homófonos $f_F(a)$ y la columna define otro conjunto de homófonos $f_C(a)$.

c) Para proceder al cifrado, se escriben los dos mensajes, uno debajo del otro, el verdadero que se llamará M y uno falso que se denominará X. El homófono que se envía como elemento del criptograma es el valor que aparece en la matriz K, en la intersección entre la fila del carácter en claro verdadero y la columna del carácter del mensaje falso.

d) Con esto, el criptograma se forma mediante el valor de las siguientes intersecciones $f_F(M_1)$ $f_C(X_1)$, $f_F(M_2)f_C(X_2)$, ..., $f_F(M_m)f_C(X_m)$, en donde f_F es la función lectura en filas y f_C la lectura en columnas.

A continuación se muestra parte de una hipotética tabla de asignación de homófonos para un cifrador con estas características.

	A	B	C	D	E	F	G	H	I	J ...
A	60	47	13	37	5	91	33	19	92	80
B	39	8	53	72	9	89	57	93	38	54

	A	B	C	D	E	F	G	H	I	J ...
C	73	1	21	94	65	10	82	58	36	18
D	12	48	2	84	20	59	3	11	55	99
E	40	88	97	26	52	71	79	35	64	56
F	14	95	66	22	83	78	16	41	34	27
G	96	85	15	69	25	51	42	76	17	23
H	4	61	28	46	100	24	98	70	67	90
I	74	29	77	86	50	62	6	43	44	32
J	7	49	68	30	45	75	63	87	31	81
...										

Luego, al cifrar el mensaje M = CEDIDA con el mensaje falso X = FIJADA, ambas palabras con sentido, usando la tabla anterior se obtiene:

C = 10 64 99 74 84 60

Observe que en estos cifrados el tamaño del texto en claro debe ser el mismo que el del texto falso.

2.4.2.3. Criptoanálisis de los cifrados por homófonos

Los criptosistemas con homófonos pueden llegar a ser extremadamente difíciles de romper, especialmente si la asignación de tales valores no sigue una lógica como en los ejemplos anteriores, en que éstos eran obtenidos a partir de un texto muy conocido. Con una gran cantidad de texto cifrado es posible encontrar algunas cadenas de números o símbolos que se repiten y, por tanto, se pueden forman digramas, trigramas y en el mejor de los casos palabras y frases completas. Si todo va bien, con un poco de suerte se podría obtener en algunos casos la tabla de homófonos o gran parte de ella.

Para los cifradores por homófonos de segundo orden, una técnica que puede dar algún fruto, también en función de que se cuente con una gran cantidad de texto cifrado, consiste en asociar los números repetidos a pares de letras de alta frecuencia, ir rellenando la matriz y, a su vez, buscar digramas, trigramas, palabras, etc., con el objeto de obtener la matriz de cifrado. Análogamente, lo mismo puede decirse para cifradores de mayor orden. No se profundizará en este tipo de criptoanálisis en este libro pues es menester contar con un texto cifrado muy grande y no tiene sentido ocupar páginas en ello. El lector interesado en las técnicas para romper estos cifradores puede consultar referencias anteriores como *David Kahn* o similares.

Ejemplo: Haciendo uso de nuestras habilidades y fuentes de información que no se van a desvelar en este momento, se ha encontrado el siguiente trozo de una tabla de homófonos, relacionada con el criptograma de 43 elementos que se indica. Encuentre los mensajes que han dado lugar al criptograma.

C = 699 289 492 124 005 693 404 017 126 559 710 590 700 258 046 124 200 705 159 200 545 581 545 644 503 388 590 219 150 041 480 727 086 346 468 603 444 013 668 077 590 100 301.

Parte de la Tabla de Homófonos (ordenada numéricamente)

1[er] carácter (Fila) 2º carácter (Columna).

005 EL	013 RE	017 EA	041 ED	046 DL	077 TC	086 AV	100 ES	124 AA
126 AP	150 TO	159 UR	200 SE	219 AT	258 OE	289 EO	301 SA	346 NU
388 TA	404 VL	444 EV	468 UE	480 GO	492 LV	503 SI	545 AN	559 PE
581 LU	590 RA	603 ML	644 EC	668 OA	693 ZE	699 PN	700 TP	705 YZ
710 ON	727 IY							

Solución: Leyendo los primeros caracteres de los homófonos del criptograma, es decir filas, se tiene:

699=PN ⇒ P	289=EO ⇒ E	492=LV ⇒ L	124=AA ⇒ A	005=EL ⇒ E
693=ZE ⇒ Z	404=VL ⇒ V	017=EA ⇒ E	126=AP ⇒ A	559=PE ⇒ P
710=ON ⇒ O	590=RA ⇒ R	700=TP ⇒ T	258=OE ⇒ O	046=DL ⇒ D
124=AA ⇒ A	200=SE ⇒ S	705=YZ ⇒ Y	159=UR ⇒ U	200=SE ⇒ S
545=AN ⇒ A	581=LU ⇒ L	545=AN ⇒ A	644=EC ⇒ E	503=SI ⇒ S
388=TA ⇒ T	590=RA ⇒ R	219=AT ⇒ A	150=TO ⇒ T	041=ED ⇒ E
480=GO ⇒ G	727=IY ⇒ I	086=AV ⇒ A	346=NU ⇒ N	468=UE ⇒ U
603=ML ⇒ M	444=EV ⇒ E	013=RE ⇒ R	668=OA ⇒ O	077=TC ⇒ T
590=RA ⇒ R	100=ES ⇒ E	301=SA ⇒ S		

De esta forma se obtiene el mensaje por filas:

M_{Fila} = PELÁEZ, VE A POR TODAS Y USA LA ESTRATEGIA NÚMERO TRES.

Leyendo los segundos caracteres de los homófonos del criptograma:

699=PN ⇒N	289=EO ⇒ O	492=LV ⇒ V	124=AA ⇒ A	005=EL ⇒ L
693=ZE ⇒ E	404=VL ⇒ L	017=EA ⇒ A	126=AP ⇒ P	559=PE ⇒ E
710=ON ⇒ N	590=RA ⇒ A	700=TP ⇒ P	258=OE ⇒ E	046=DL ⇒ L
124=AA ⇒ A	200=SE ⇒ E	705=YZ ⇒ Z	159=UR ⇒ R	200=SE ⇒ E
545=AN ⇒ N	581=LU ⇒ U	545=AN ⇒ N	644=EC ⇒ C	503=SI ⇒ I
388=TA ⇒ A	590=RA ⇒ A	219=AT ⇒ T	150=TO ⇒ O	041=ED ⇒ D
480=GO ⇒ O	727=IY ⇒ Y	086=AV ⇒ V	346=NU ⇒ U	468=UE ⇒ E
603=ML ⇒ L	444=EV ⇒ V	013=RE ⇒ E	668=OA ⇒ A	077=TC ⇒ C
590=RA ⇒ A	100=ES ⇒ S	301=SA ⇒ A		

Ahora se obtiene el mensaje por columna:
$M_{columna}$ = NO VALE LA PENA PELÁEZ, RENUNCIA A TODO Y VUELVE A CASA.

En el ejemplo anterior, incluso conociendo de qué va el affaire de nuestro querido amigo Peláez, no será posible dilucidar cuál de los dos mensajes que se han criptoanalizado es el verdadero y cuál el falso, salvo que se conozca la clave de lectura en la tabla. Por lo tanto, nuestro esfuerzo en romper la tabla de homófonos nos ha llevado a un callejón sin salida: tener mucho texto con sentido pero asociado a una gran incertidumbre sobre la información que contiene y la veracidad de la misma. En otras palabras y aunque parezca un sarcasmo, después de rompernos la cabeza, no se tiene nada.

2.4.3 Cifradores por sustitución monográmica polialfabeto

Los criptosistemas monoalfabéticos por sustitución y los de transposición o permutación presentan una gran debilidad al poder romperse en muchos casos los criptogramas aplicando unas técnicas sencillas de estadísticas del lenguaje. Esta debilidad está asociada directamente al hecho de que en el primero de ellos la sustitución se realizaba siempre con un único carácter del alfabeto de cifrado y, en el segundo, a que las letras del criptograma eran exactamente las mismas que las del texto en claro y sólo se han roto algunas adyacencias de caracteres. Ambos escenarios entregan una ayuda inapreciable al criptoanalista. Para solucionar estos problemas, aparecen los cifradores por sustitución polialfabéticos que, como su nombre lo indica, usan más de un alfabeto para cifrar.

Los algoritmos de sustitución polialfabética tienen por objeto producir una distribución plana de la frecuencia relativa de los caracteres en el criptograma, de una manera similar a la técnica de los homófonos. Para ello utilizan sustituciones múltiples de forma que en un texto largo se combinan las altas frecuencias de algunos caracteres con otros de menor frecuencia. En otras palabras, si por ejemplo la letra A, de alta frecuencia en el lenguaje castellano, se cifra algunas veces como la letra O y otras veces como la letra J (una de alta frecuencia y la otra de baja) y lo mismo ocurre para la letra W, de baja frecuencia en el lenguaje, el efecto final es suavizar la mencionada distribución de frecuencia de todos los caracteres del criptograma.

La técnica anterior consiste en aplicar dos o más alfabetos de cifrado de forma que cada uno de ellos sirva para cifrar los caracteres del texto en claro, dependiendo de la posición relativa de éstos en dicho texto. Por ejemplo, si se utiliza un alfabeto A_1 para cifrar los caracteres de las posiciones impares del mensaje y otro alfabeto A_2 para los caracteres en posiciones pares, entonces en un texto lo suficientemente extenso se tendrá que, aproximadamente, sólo en la mitad de las ocasiones las letras repetidas del texto en claro se repiten como un mismo elemento c_i en el criptograma, lográndose por tanto el efecto de enmascaramiento de la distribución de frecuencia de los monogramas característica del lenguaje.

Cifrador de Alberti

Un ejemplo famoso de este tipo de cifrado se encuentra en el cifrador de *Alberti*. En el siglo XVI *Leon Battista Alberti* presenta un manuscrito en el que describe un disco cifrador con el que es posible cifrar textos sin que exista una correspondencia única entre el alfabeto del mensaje y el alfabeto de cifrado como en los casos analizados anteriormente. Con este sistema, cada letra del texto en claro podía ser cifrada con un carácter distinto dependiendo esto de una clave secreta. Como se aprecia en la siguiente figura, el disco de *Alberti* presenta en su círculo exterior los 20

caracteres del latín, esto es, los mismos del alfabeto castellano excepto las letras H, J, Ñ, K, U, W e Y, y se incluyen los números 1, 2, 3 y 4 para códigos especiales. Por su parte, en el disco interior aparecen todos los caracteres del latín además del signo & y las letras H, K e Y. Al ser 24 los caracteres representados en cada disco, es posible definir hasta 24 sustituciones diferentes; es decir, dependiendo de la posición del disco interior la cantidad máxima de alfabetos de cifrado es igual a 24. Luego, para cifrar un mensaje, una vez establecida la correspondencia entre caracteres de ambos discos o, lo que es lo mismo, el alfabeto de cifrado, se repasa letra a letra el texto en claro del disco exterior y se sustituye cada una de ellas por la letra correspondiente del disco interior.

La innovación que supone este sistema consiste en que el alfabeto de sustitución puede ser cambiado durante el proceso de cifrado, por ejemplo cada k caracteres, simplemente girando el disco interior y por tanto utilizando otro alfabeto de sustitución.

Ejemplo: Cifre con el disco de *Alberti* de la Figura 3, siendo su posición inicial la de coincidencia entre el número 1 del disco exterior y el signo & del disco interior, el siguiente mensaje:

> M = EL DISCO DE ALBERTI ES EL PRIMER CIFRADOR POLIALFABÉTICO CONOCIDO.

Solución: Se desplaza el disco interior dos espacios en el sentido de las agujas del reloj y se lee el carácter cifrado en el disco interior bajo el carácter correspondiente del texto en claro del disco exterior, obteniéndose:

> C = VA EOSMP EV HARVXFO VS VA BXOIVX MOLXHEPX BPAOHALHRVFOMP MPYPMOEP.

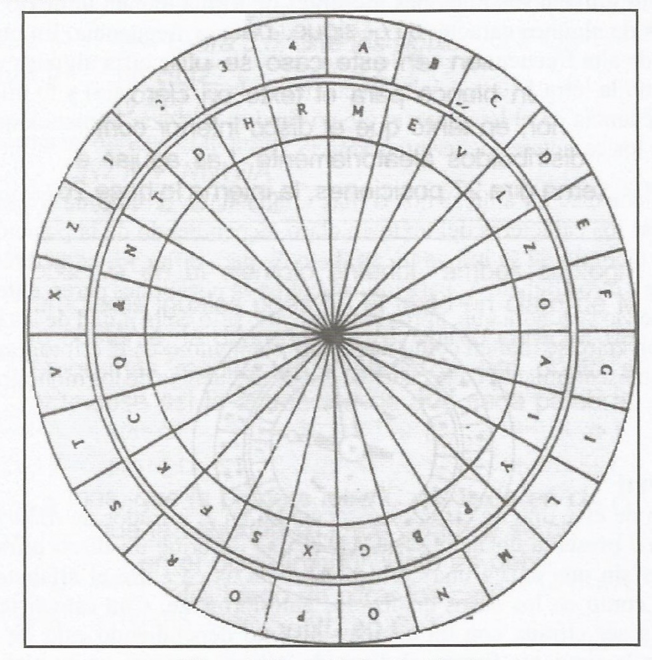


Figura 3. Disco cifrador de *Alberti*.

Cifrador de Wheatstone

El criptógrafo de *Wheatstone* -según un invento de *Decius Wadsworth* desarrollado en 1817- sigue, básicamente, el mismo algoritmo de cifra que el de *Alberti*. Ahora bien, en este caso se utiliza el alfabeto inglés de 26 caracteres más el espacio en blanco para el texto en claro, representado de forma ordenada en el disco exterior, en tanto que el disco interior contiene solamente los 26 caracteres del lenguaje distribuidos aleatoriamente. Las agujas están engranadas de forma que cuando la externa gira 27 posiciones, la interna lo hace 26.

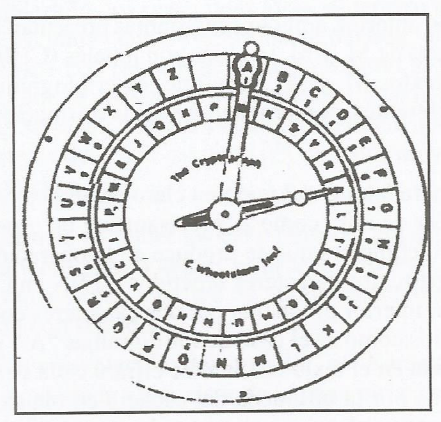


Figura 4. Máquina de cifrar de *Wheatstone*.

El método de cifra consiste en hacer girar la aguja externa en el sentido de las manecillas del reloj hasta hacer coincidir cada letra del texto en claro con la letra del disco externo y apuntar el carácter correspondiente que aparece en el círculo interior, incluso para el espacio en blanco. Observe que por la relación de giro de las agujas, éstas se van separando una posición o letra por cada vuelta, de forma que el alfabeto de cifrado será diferente cuando se cumpla cualquiera de estas tres condiciones:

a) Que se termine una palabra del texto en claro y por tanto se dé un giro completo de la aguja mayor al buscar el espacio en blanco.

b) Que aparezcan letras repetidas y sea necesario dar toda una vuelta completa al buscar la segunda. No obstante, según los autores, en este caso es posible también omitir cifrar la letra repetida o bien cifrar ambas como una única letra poco usual, por ejemplo la letra Q.

c) Que las letras de una palabra no vengan en orden alfabético. Es decir, si se cifra la palabra CELOS no se alcanza a dar la vuelta completa al disco exterior, en tanto que la palabra MUJER implica dos vueltas y HOMBRE significa tres. No trate de encontrar ningún mensaje subliminal en estas tres palabras y sus vueltas.

La importancia de este cifrador está en que cada una de las palabras del mensaje influye en la forma en que se cifran las siguientes, una propiedad muy interesante y que precisamente utilizarán los cifradores modernos, sencillamente definiendo el concepto de palabra como bloque de bits para la cifra y aplicando lo que se denomina cifrado con encadenamiento de bloques.

Ejemplo: Con la máquina de cifrar de *Wheatstone* y suponiendo la posición inicial indicada en la Figura 4, cifre los siguientes mensajes:

M_1 = CHICA FELIZ.

M_2 = CHICO FELIZ.

Solución:

C_1 = TUNZT T NNWIA.

C_2 = TUNZW L UUPCZ.

Como se observa en el ejemplo anterior, ambos criptogramas presentan los cuatro primeros caracteres iguales pues en el texto en claro de M_1 y M_2 también son iguales (CHIC). No obstante, la diferencia en el quinto carácter de los textos M_1 y M_2, hace que los criptogramas resultantes a partir de ese punto sean completamente diferentes, comprobándose así la afirmación de que cada palabra influye en el cifrado de la siguiente.

Observe, además, que la primera letra C del texto en claro en ambos casos se cifra como T, en tanto que la segunda vez que aparece se cifra como Z, precisamente un espacio hacia delante en el disco interior. Esto es debido al giro completo que se produce en la operación de cifra luego de cifrar los caracteres C, H e I. Por otra parte, los caracteres repetidos TTNN en C_1 y UU en C_2 se deben a una revolución completa del disco interior producida por dos caracteres contiguos en el texto en claro y que están separados 26 espacios como es el caso de los digramas "A " y "FE". Por último, apréciese que una misma palabra repetida en el texto en claro se cifrará cada vez con un alfabeto distinto por la rotación completa producida por la búsqueda del espacio en blanco. Por ejemplo el mensaje M = TORA TORA, palabra secreta usada como clave por los japoneses en el ataque a Pearl Harbor y cuyo significado es tigre, se cifrará como C = XWQT Z KQBG.

Cifrador de Bazeries

El cifrador de *Étienne Bazeries*, criptólogo francés nacido a finales del siglo XIX, está basado en el cifrador de ruedas de *Jefferson*, inventado unos 100 años antes por *Thomas Jefferson* reconocido como el padre de la criptografía americana. El criptógrafo mostrado en la figura siguiente consta de 20 discos, cada uno de ellos con 25 letras en su circunferencia, de forma que la clave se establece sobre la generatriz del cilindro, determinándose 25 alfabetos diferentes. Su funcionamiento es el siguiente: para cifrar el mensaje, primero se divide éste en bloques de 20 letras, procediendo luego a su colocación en forma longitudinal en la línea del visor. El criptograma que se envía puede ser cualquiera de las 25 líneas, también llamadas generatrices del cilindro. Por ejemplo, si se elige la generatriz de distancia +2 en la Figura 5, el mensaje M = JE SUIS INDECHIFFRABLE del visor se cifraría como C = LOVS PQUU TPUKEJHHCFDA.

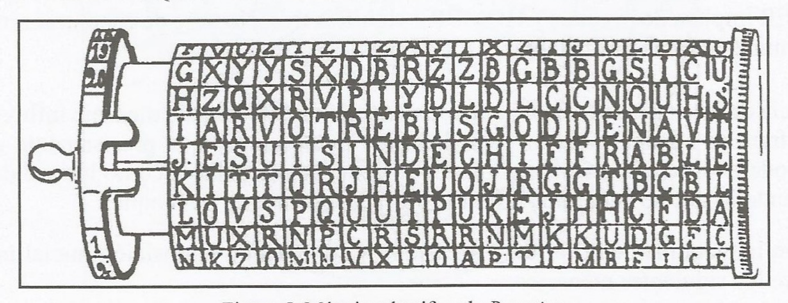


Figura 5. Máquina de cifrar de *Bazeries*.

Se puede elegir la misma distancia a la generatriz en la cual se lee el criptograma para todo el bloque o bien cambiar ésta en cada bloque o elemento del bloque, de forma que el número de combinaciones o alfabetos distintos en vez de ser solamente 25 podría crecer hasta el factorial de 25, un valor verdaderamente alto. Uno de estos posibles alfabetos podría ser elegir una secuencia de distancias, una vez introducido el mensaje en el visor, igual a -1,-2,-2,-1,1,2,2,1,-1,-2,-2,-1,1,2,2,1,-1,-2,-2,-1. Es decir, una vez se tenga el mensaje en claro en el visor, se envía como primer carácter del criptograma el que, en la misma columna, está desplazado una posición hacia atrás en el anillo; como segundo el que está desplazado dos posiciones atrás, el tercero también dos posiciones atrás, el cuarto una posición atrás, el quinto una posición hacia delante, el sexto dos adelante, etc., de manera que el criptograma forma una especie de zigzag en torno al texto en claro, sin transmitir ningún carácter de éste puesto que la posición 0 no se encuentra en la secuencia indicada. Como es fácil observar, dicha secuencia sería la clave del sistema y, en este caso, su valor máximo sería igual todas las posible permutaciones es decir $25! = 1{,}55*10^{25}$, un valor muy grande aunque el sistema de cifra sería engorroso y poco práctico.

La operación de descifrado consiste en poner los caracteres del criptograma en el visor y buscar en alguna de las líneas el mensaje en claro o seguir el proceso inverso al comentado anteriormente. Como los bloques de criptograma tienen longitud de veinte caracteres, es prácticamente imposible que exista más de una solución con sentido.

Ejemplo: Considerando una representación del cifrador de *Bazeries* como la que se indica a continuación, cifre el mensaje mostrado en el visor de la generatriz 11 del disco:

M = INTENTA ROMPER LA CIFRA.

a) Con una distancia constante de +3 espacios.

b) Con la secuencia S de distancia de cifrado indicada:

S = 0,1,2,1,0,-1,-2,-1,0,1,2,1,0,-1,-2,-1,0,1,2,1.

Fila Disco	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0
....																				
7	V	A	M	W	W	U	I	O	P	S	S	A	H	K	L	V	C	D	U	Q
8	M	O	J	F	J	K	L	M	A	C	H	Y	E	D	X	Z	G	Q	I	U
9	D	B	W	Q	D	S	**B**	D	Q	T	F	D	S	F	**D**	X	E	W	Q	G
10	A	W	K	Y	H	**M**	P	**E**	S	H	U	K	P	**U**	O	**E**	S	J	K	D
11	**I**	N	T	E	**N**	T	A	R	**O**	M	P	E	**R**	L	A	C	**I**	F	R	A
12	R	**G**	I	**S**	X	F	W	G	B	**A**	N	**L**	F	M	J	H	A	**A**	S	**H**
13	U	I	**D**	V	C	R	I	I	Z	D	**D**	C	Z	A	K	I	M	B	**L**	X
14	K	L	B	O	T	Z	H	Y	L	V	C	O	N	D	W	A	L	M	V	Z
15	W	H	S	K	L	P	O	U	I	E	E	P	D	N	C	G	Q	E	O	B
......																				

Solución: a) C = KLBOTZHYLVCONDWALMVZ.

b) Según la secuencia indicada, se toman caracteres consecutivos de las líneas 11,12,1 3,12,11,10,9,10,11,12,13,12,11,10,9,10,11,12,13,12 que se encuentran subrayados. C = IGDSNMBEOADLRUDEIALH.

Todos los sistemas comentados y muchísimos otros que se desarrollaron paralelamente en Europa, América y Asia, han sido criptoanalizados incluso sin contar con la ayuda de equipos informáticos. No obstante, la discusión de su criptoanálisis está fuera del objetivo de este libro, por lo que al lector interesado en tales temas históricos se le remite nuevamente al libro de Kahn y publicaciones similares.

2.4.3.1. Cifradores polialfabéticos periódicos

Los sistemas por sustitución polialfabética tienen, por lo general, un período que vendrá dado por la longitud de la clave de cifrado. La única excepción se encuentra en los denominados cifradores polialfabéticos de clave continua, siendo un ejemplo característico el cifrador de *Vernam*. Los cifradores de clave continua poseen una clave tanto o más larga que el texto en claro por lo que serán cifradores no periódicos.

Luego, si dependiendo de la longitud de la clave se tienen *d* alfabetos de cifrado, habrá una periodicidad en los elementos del criptograma indicada por la siguiente cadena:

$$C = C_1 \ldots C_d C_{d+1} \ldots C_{2d} C_{2d+1} \ldots C_{3d} \ldots$$

O lo que es lo mismo, si f: A → C es una aplicación de correspondencia del alfabeto A del texto en claro con el alfabeto de cifrados C_i, con $1 \leq i \leq d$, entonces se tiene que el mensaje $M = M_1 \ldots M_d M_{d+1} \ldots M_{2d} \ldots$, se cifrará repitiendo la secuencia $f_1 \ldots f_d$ cada d caracteres. Por lo tanto:

$$E_K(M) = f_1(M_1) \ldots f_d(M_d) f_1(M_{d+1}) \ldots f_d(M_{2d}) f_1(M_{2d+1}) \ldots$$

Cifrador de Vigenère

El cifrador polialfabético más conocido es el sistema de *Vigenère*, así denominado en honor al criptólogo francés *Blaise de Vigenère* (1523-1596). El sistema utiliza el mismo método que el cifrador del *César*, esto es una sustitución monográmica por desplazamiento de k caracteres en el texto, con la diferencia de que dicho desplazamiento viene indicado por el valor numérico asociado a uno de los caracteres de una clave que se escribe cíclicamente bajo el mensaje como se indica a continuación:

```
TEXTO: E N U N L U G A R D E L A M A N C H A D E C U Y O N O M B R E ...
CLAVE: C E R V A N T E S C E R V A N T E S C E R V A N T E S C E R V ...
```

Según lo anterior, la clave utilizada será CERVANTES y tendrá una periodicidad igual a 9, pues son los caracteres que forman esta palabra. Luego, al primer carácter del texto en claro (E) se le aplica un desplazamiento equivalente al primer carácter de la clave (C), dando lugar a E + C = (4 + 2) mod 27 = 6 = G; el segundo carácter (N) se cifra sumándolo con el segundo carácter de la clave (E), N + E = (13 + 4) mod 27 = 17 = Q, etc. El resultado final será el criptograma: C = GQMIL HZEKF ICVMN GGZCH VXULI. Compruebe este resultado.

Ejemplo: Aplicando relaciones de congruencia como las indicadas en el párrafo anterior, cifre el siguiente mensaje según el método de *Vigenère*.

M = DESASTRE NUCLEAR EN MURUROA. Clave K = SOS.

Solución:

```
D E S A S T R E N U C L E A R E N M U R U R O A
S O S S O S S O S S O S S O S S O S S O S S O S
```

$D+S = (3+19) \bmod 27 = 22 \Rightarrow V$

$E+O = (4+15) \bmod 27 = 19 \Rightarrow S$

$S+S = (19+19) \bmod 27 = 11 \Rightarrow L$

$A+S = (0+19) \bmod 27 = 19 \Rightarrow S$

$S+O = (19+15) \bmod 27 = 7 \Rightarrow H$

$T+S = (20+19) \bmod 27 = 12 \Rightarrow M$

$R+S = (18+19) \bmod 27 = 10 \Rightarrow K$

$E+O = (4+15) \bmod 27 = 19 \Rightarrow S$

$N+S = (13+19) \bmod 27 = 5 \Rightarrow F$

$U+S = (21+19) \bmod 27 = 13 \Rightarrow N$

$C+O = (2+15) \bmod 27 = 17 \Rightarrow Q$

$L+S = (11+19) \bmod 27 = 3 \Rightarrow D$

$E+S = (4+19) \bmod 27 = 23 \Rightarrow W$

$A+O = (0+15) \bmod 27 = 15 \Rightarrow O$

$R+S = (18+19) \bmod 27 = 10 \Rightarrow K$

$E+S = (4+19) \bmod 27 = 23 \Rightarrow W$

$N+O = (13+15) \bmod 27 = 1 \Rightarrow B$

$M+S = (12+19) \bmod 27 = 4 \Rightarrow E$

$U+S = (21+19) \bmod 27 = 13 \Rightarrow N$

$R+O = (18+15) \bmod 27 = 6 \Rightarrow G$

$U+S = (21+19) \bmod 27 = 13 \Rightarrow N$

$R+S = (18+19) \bmod 27 = 10 \Rightarrow K$

$O+O = (15+15) \bmod 27 = 3 \Rightarrow D$

$A+S = (0+19) \bmod 27 = 19 \Rightarrow S$

Luego, se obtiene el criptograma: C = VSLSH MKSFN QDWOK WBENG NKDS.

Observe que letras repetidas del texto en claro se cifran de forma distinta, dependiendo de su posición relativa respecto a la clave. Es el caso de la letra E que se cifra dos veces como S al coincidir con la letra O de la clave y dos veces como W cuando la letra de la clave es S. Algo similar ocurre con las letras A, N y R y no así con la U que se cifra tres veces como N. Por otra parte, una letra repetida del criptograma puede provenir de caracteres distintos del texto en claro. Es el caso de la letra D que proviene de los caracteres L y O del mensaje. Las observaciones anteriores pueden generalizarse teniendo en cuenta el número de alfabetos utilizados en función de la clave. En nuestro ejemplo, si bien la clave SOS implica una periodicidad igual a tres, solamente se utilizan dos alfabetos, el correspondiente a la letra S y el de la letra O.

CLAVE		TEXTO EN CLARO																										
		0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	1	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	2	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	3	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	4	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	5	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	6	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	7	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	8	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	9	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	10	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	11	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K

M	12	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	13	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
Ñ	14	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
O	15	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ
P	16	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O
Q	17	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P
R	18	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q
S	19	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R
T	20	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S
U	21	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T
V	22	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U
W	23	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V
X	24	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W
Y	25	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X
Z	26	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y

Tabla 2. Tabla de cifrado de *Vigenère*

El algoritmo de *Vigenère* utiliza permutaciones para cifrar los caracteres del texto en claro con una clave. Si se extiende el número de permutaciones hasta su límite superior, en nuestro caso 27, se obtiene la denominada Tabla de *Vigenère*.

Para cifrar un texto utilizando la Tabla de *Vigenère* se procede de la siguiente manera: posicionarse en el carácter del texto en claro a cifrar en la primera fila de la tabla y buscar la letra de la clave en cuestión en la primera columna de la tabla. El elemento C_i del criptograma será la letra de la retícula de intersección entre fila y columna. Por ejemplo la letra M cifrada con la clave O nos dará el criptograma A. Compruebe que al mismo resultado se llega si se cifra el texto en claro O con la clave M. La primera fila, la de la clave A, corresponde a la del texto en claro pues es lo que se obtiene al aplicar un desplazamiento igual a cero.

Ejemplo: Utilizando la Tabla de *Vigenère* y la clave K = WINDOWS, cifre el siguiente mensaje:

M = MARIPURI, APAGA ESE ORDENADOR.

Solución:

```
M A R I P U R I A P A G A E S E O R D E N A D O R
W I N D O W S W I N D O W S W I N D O W S W I N D
```

Buscando en la tabla, se obtiene el criptograma:
C = IIELE QKEIC DUWWO MBURA FWLBU.

En el ejemplo anterior la clave tenía longitud 7 aunque sólo se han usado de 6 alfabetos de cifrado, los de las letras no repetidas de la clave W, I, N, D, O, S.

Puesto que la clave está formada por un conjunto de d caracteres, $K = k_1 \ldots k_d$, en donde k_i ($i = 1,\ldots d$) entrega la cantidad de desplazamiento del alfabeto i-ésimo, la función de transformación de *Vigenère* para cifrar vendrá dada por:

$$C_i = E_{ki}(M_i) = (M_i + k_i) \bmod n$$

Para realizar la operación de descifrado con la tabla se procede de manera inversa. En la fila del carácter i-ésimo de la clave, posicionarse en la retícula de la letra del criptograma; hecho esto, se sube por esta columna hasta la fila primera del texto en claro y se lee el carácter. Por ejemplo, si el elemento C_i es la letra G y el elemento k_i es la letra Ñ, el resultado será el texto en claro S.

Ejemplo: En la sala de mandos se recibe el siguiente criptograma que se sabe ha sido cifrado mediante *Vigenère* con la clave TORA. Se pide descifrarlo.

C = RODAF DLOIG UEGOR TTQRR JSRRE VRRUD J.

Solución: Aplicando el método comentado, se obtiene el inquietante mensaje:

M = YAMAMOTO ORDENA ATACAR PEARL HARBOR.

De acuerdo con la operación de cifra, la función de descifrado de *Vigenère* deberá utilizar el inverso del desplazamiento aplicado, dando lugar a la expresión:

$$M_i = D_{ki}(C_i) = (C_i - k_i) \bmod n$$

Por ejemplo, para el primer elemento del criptograma del ejemplo anterior (R), con letra de clave (T) se tiene: (R-T) mod 27 = (18-20) mod 27 = -2 mod 27 = 25 = Y. Continúe Ud. con el resto del criptograma.

Cifrador autoclave

Es una variante del algoritmo de *Vigenère*, conocido también como Segundo Cifrado de *Vigenère*, cuya característica radica en que se cifra el mensaje con una clave que consiste en el mismo mensaje al que se le añade al comienzo una clave denominada primaria. Luego, la secuencia de clave será tan larga como el propio mensaje. Por ejemplo, si utilizando la clave K = MARKETING se desea cifrar el mensaje M = YA ES PRIMAVERA EN EL CORTE INGLÉS, usando la Tabla de *Vigenère* se obtiene el siguiente criptograma:

```
M = Y A E S P R I M A V E R A E N E L C O R T E I N G L E S
K = M A R K E T I N G Y A E S P R I M A V E R A E N E L C O
C = K A V C T L P Y G T E V S T E M W C K V L E M Z K V G H
```

La operación de descifrado, conociendo la clave MARKETING es igual que en *Vigenère*. Esto es, se descifran los nueve primeros caracteres como se indica:

$K - M \Rightarrow (10 - 12) \bmod 27 = 25 \quad \Rightarrow Y$
$A - A \Rightarrow (0 - 0) \bmod 27 = 0 \quad \Rightarrow A$
$V - R \Rightarrow (22 - 18) \bmod 27 = 4 \quad \Rightarrow E$
$C - K \Rightarrow (2 - 10) \bmod 27 = 19 \quad \Rightarrow S$
$T - E \Rightarrow (20 - 4) \bmod 27 = 16 \quad \Rightarrow P$
$L - T \Rightarrow (11 - 20) \bmod 27 = 18 \quad \Rightarrow R$
$P - I \Rightarrow (16 - 8) \bmod 27 = 8 \quad \Rightarrow I$
$Y - N \Rightarrow (25 - 13) \bmod 27 = 12 \quad \Rightarrow M$
$G - G \Rightarrow (6 - 6) \bmod 27 = 0 \quad \Rightarrow A$

Con este método podrá descifrar sólo los primeros 9 caracteres del criptograma KAVCTLPYG correspondientes a la clave primaria MARKETING. Para continuar descifrando, debe hacer uso de los 9 caracteres ya descifrados (YAESPRIMA) que, según el método, irán a continuación de dicha clave. Por lo tanto, se descifra ahora el criptograma TEVSTEMWC con la clave YAESPRIMA y así sucesivamente hasta obtener el mensaje original. Aunque impresione más que otras técnicas de cifra, el secreto de este criptosistema reside únicamente en el de la clave. Conocida ésta, el criptoanálisis es elemental.

Ejemplo: Si la clave usada en un cifrador autoclave es PIZZA, descifre el siguiente criptograma.

C = SWMCE HHGDI OLXCV OMSGC WXQVO MSGKX TSQDT MEFL

Solución:

Bloque 1: SWMCE Clave: PIZZA $\Rightarrow M_1$ = DONDE
Bloque 2: HHGDI Clave: DONDE $\Rightarrow M_2$ = ESTAE
Bloque 3: OLXCV Clave: ESTAE $\Rightarrow M_3$ = LSECR
Bloque 4: OMSGC Clave: LSECR $\Rightarrow M_4$ = ETOEL
Bloque 5: WXQVO Clave: ETOEL $\Rightarrow M_5$ = SECRE
Bloque 6: MSGKX Clave: SECRE $\Rightarrow M_6$ = TOEST
Bloque 7: TSQDT Clave: TOEST $\Rightarrow M_7$ = AENLA
Bloque 8: MEFL Clave: AENL $\Rightarrow M_8$ = MASA

Luego, incluyendo los signos de puntuación, el mensaje es (incluido signos):
M = ¿DÓNDE ESTÁ EL SECRETO? EL SECRETO ESTÁ EN LA MASA.

Cifrador de Beaufort

En 1710, *Giovanni Sestri*, basado en el método de cifra de *Vigenère*, propone un algoritmo simétrico que sirve tanto para cifrar como para descifrar. El invento del cifrador en cuestión finalmente se le atribuye al inglés *Sir Francis Beaufort*, amigo de *Sestri*, y recibe precisamente el nombre de cifrador de *Beaufort*. Eso sí que es un amigo. La sustitución empleada en este cifrador sigue la siguiente expresión:

$$C_i = E_{ki}(M_i) = (k_i - M_i) \bmod n$$

CLAVE TEXTO EN CLARO

		0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	0	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B
B	1	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C
C	2	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D
D	3	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E
E	4	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F
F	5	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G
G	6	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H
H	7	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I
I	8	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J

J	9	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K
K	10	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L
L	11	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M
M	12	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N
N	13	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ
Ñ	14	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P	O
O	15	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q	P
P	16	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R	Q
Q	17	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S	R
R	18	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T	S
S	19	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U	T
T	20	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V	U
U	21	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W	V
V	22	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X	W
W	23	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y	X
X	24	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z	Y
Y	25	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z
Z	26	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A

Tabla 3. Tabla de cifrar *Beaufort* para el lenguaje castellano.

Por lo tanto, se invierte el orden de las letras del alfabeto A y luego se les aplica un desplazamiento hacia la derecha de $(k_i + 1)$ posiciones. Esta afirmación puede comprobarse aplicando la siguiente congruencia:

$$E_{ki}(A) = (k_i - A) \bmod n = [(n-1) - A + (k_i + 1)] \bmod n$$

Ejemplo: Usando la Tabla de *Beaufort* con clave K = ULTIMATUM, cifre el mensaje:

M = ESTO ES LA GUERRA SEÑORES.

Solución: Siguiendo la tabla de la Figura 1.23 se obtiene:

C = QSATI IJUGA HCQMI PHXDH B.

Por la operación de sustitución empleada es posible que, a diferencia del de *Vigenère*, un carácter se cifre con su valor en claro con una clave distinta de la letra A, como es el caso en el ejemplo anterior en que el segundo (S) y el noveno (G) caracteres se cifran en claro, con las claves (L) y (M), respectivamente. La operación de descifrado es la misma que la de cifrado, con la excepción que en vez de texto claro M se cuenta ahora con texto cifrado C, esto es:

$$M_i = D_{ki}(C_i) = (k_i - C_i) \bmod n$$

Por ejemplo, para los cinco primeros caracteres del ejemplo anterior (QSATI) se obtiene:

$k_1 = U; C_1 = Q \Rightarrow M_1 = (21-17) \bmod 27 = 4 \Rightarrow M_1 = E$
$k_2 = L; C_2 = S \Rightarrow M_2 = (11-19) \bmod 27 = 19 \Rightarrow M_2 = S$
$k_3 = T; C_3 = A \Rightarrow M_3 = (20-0) \bmod 27 = 20 \Rightarrow M_3 = T$
$k_4 = I; C_4 = T \Rightarrow M_4 = (8-20) \bmod 27 = 15 \Rightarrow M_4 = O$
$k_5 = M; C_5 = I \Rightarrow M_5 = (12-8) \bmod 27 = 4 \Rightarrow M_5 = E$

A igual resultado puede llegarse si se utiliza la Tabla de *Beaufort* para descifrar. De forma similar al método de *Vigenère*, habrá que posicionarse en la fila correspondiente al carácter de la clave y buscar la retícula en la que aparezca el elemento cifrado, su proyección a la fila superior del texto en claro nos entrega el carácter del mensaje. Observe que en este caso, la fila de desplazamiento 0 (letra A) no se corresponde con la del texto en claro como sucedía con *Vigenère*.

Variante del cifrador de Beaufort

Si se modifica la función de cifrado E_{ki} de forma que se transforme en $E_{ki}(M_i) = (M_i - k_i) \bmod n$, el sistema se conoce como variante de *Beaufort*. Esto es equivalente a cifrar con *Vigenère* siendo la clave $(n - k_i)$ lo que puede demostrarse a partir de la siguiente congruencia para un alfabeto A:

$$(A - k_i) \bmod n = [A + (n - k_i)] \bmod n$$

Al ser el inverso del algoritmo de *Vigenère*, este cifrador se puede utilizar para descifrar criptogramas obtenidos con *Vigenère* y viceversa. Algo obvio por lo demás.

2.4.3.2. Criptoanálisis de los cifrados polialfabéticos periódicos

Al utilizar más de un alfabeto, el número de combinaciones de la clave crecerá y también lo hará su entropía y distancia de unicidad. Para un cifrador polialfabético como *Vigenère*, la distancia de unicidad vendrá dada por el número total de combinaciones usadas para sustituciones simples; esto es, si para cada sustitución simple monoalfabeto hay n posibles claves, entonces al utilizar d sustituciones existirán n^d claves posibles.

Ejemplo: Si el alfabeto de claves son las letras A, B, C y D:

a) ¿Cuántas claves de dos elementos se pueden formar?

b) Encuentre la distancia de unicidad del cifrador de *Vigenère* para el lenguaje castellano.

c) ¿Cuál es su valor para una clave de longitud 10?

Solución:

a) Existirán $n^d = 4^2 = 16$ combinaciones posibles. Para este alfabeto de cuatro letras, serán claves: AA, AB, AC, AD, BA, BB, BC, BD, CA, CB, CC, CD, DA, DB, DC y DD.

b) $N = H(K)/D = \log_2 (n^d)/D = d * \log_2 n/D = d * \log_2 27/3{,}4$.

c) Si $d = 10$ entonces $N \approx 10 * 1{,}4 \approx 14$ caracteres.

Luego, para romper un cifrado polialfabético se necesitará mucho más texto cifrado que en uno monoalfabético, tantas veces como el valor de su período, puesto que en aquel caso la cantidad de texto cifrado necesaria venía dada por $\log_2 n/D$.

Método de Kasiski

Se va a profundizar en la característica de periodicidad de los cifradores polialfabéticos. Suponga que, por algún método aún desconocido por Ud., se logra adivinar que la longitud de la clave es igual

a seis caracteres. Si el criptograma C está formado por una cadena de m caracteres $C_1C_2...C_{m-1}C_m$, puede escribirlo en un formato de seis columnas como se indica:

$$
\begin{array}{cccccc}
C_1 & C_2 & C_3 & C_4 & C_5 & C_6 \\
C_7 & C_8 & C_9 & C_{10} & C_{11} & C_{12} \\
C_{13} & C_{14} & C_{15} & C_{16} & C_{17} & C_{18} \\
C_{19} & C_{20} & C_{21} & C_{22} & C_{23} & C_{24} \\
\ldots\ldots & & & & & \\
C_{m-5} & C_{m-4} & C_{m-3} & C_{m-2} & C_{m-1} & C_m
\end{array}
$$

Recordando el método de cifrado de *Vigenère* con una clave de seis caracteres, cada una de las seis columnas corresponderá a la cifra con el mismo elemento de la clave. Esto es, los caracteres de una misma columna se corresponden con los de un cifrado monoalfabético por desplazamiento puro dado por el valor del elemento i-ésimo de clave. Esto nos va a indicar que dos o más caracteres iguales en una columna se deberán a caracteres iguales del texto en claro que, evidentemente, se han cifrado con la misma letra de clave. Además, como el lenguaje castellano presenta una alta redundancia es posible que poligramas característicos tales como ando, endo, ada, ido, ado, ica, ita, ción, mente y muchos otros sean cifrados con la misma cadena de clave, originando cadenas de texto cifrado repetidas en el criptograma.

La probabilidad de que se den estas repeticiones de cadenas será menor que en un cifrador monoalfabético; no obstante, una cadena grande de caracteres repetidos es muy poco probable que aparezca por puro azar. De hecho, trigramas y tetragramas repetidos más de una vez en el criptograma indican una alta probabilidad de que la distancia entre tales cadenas sea un múltiplo de la clave utilizada para cifrar. Este principio fue observado por el criptólogo alemán *Friedrich W. Kasiski* en 1860 con lo que el método lleva su nombre. En otras palabras, si una clave tiene L caracteres, sólo hay L formas diferentes de posicionar dicha clave sobre la o las palabras en el texto en claro.

Esto es, si la clave es NECIO, una repetición típica de cuatro letras como podría ser ando (e.g. comprobando, contrabando, bando, bandolero, cantando, abandono, etc.) podrá cifrarse solamente de las siguientes cinco formas ANDO+NECI, ANDO+ECIO, ANDO+CION, ANDO+IONE y ANDO+ONEC.

Luego, para este caso, puede esperarse que un grupo de caracteres que aparecen más de L veces en el texto en claro hayan sido cifrados al menos dos veces en la misma posición de la clave y dichas ocurrencias se cifrarán de forma idéntica. Veamos un ejemplo sencillo. Si el mensaje es el famoso monólogo de Hamlet y se cifra mod 27 según el método de *Vigenère* con la clave K = HAM se tiene:

```
M = T O B E O R N O T T O B E T H A T I S T H E ...
K = H A M H A M H A M H A M H A M H A M H A M H ...
C = A O N L O D T O F A O N L T S H T T Z T S L ...
```

La cadena o secuencia de caracteres AONL que aparece dos veces en el criptograma con una separación igual a 9 espacios, sugiere que el período de la clave sea igual a 3 ó 9. Además se encuentra la cadena TS separada por 6 espacios lo que confirmaría que el período es igual a 3. Con

algo más de texto y el uso de las estadísticas del lenguaje como se verá más adelante, se podrá determinar que la clave de este ejemplo tiene efectivamente una longitud de 3 caracteres y que éstos son precisamente HAM.

La mejor manera de explicar el método de *Kasiski* es mediante un ejemplo detallado. Suponga el siguiente criptograma de 404 caracteres que se indica mostrado en grupos de cinco.

PB**VR**Q VICAD SKAÑS DETSJ PSIED B**GGMP** SLRPW RÑPW**Y EDS**DE ÑDRDP

CRCPQ MNPWK UBZVS FN**VR**D MTIPW UEQVV CBOVN UEDIF QLONM WNU**VR**

SEIKA ZYEAC E**YEDS** ETFPH LBHGU ÑESOM EHLBX VAEEP UÑELI SEVEF

WHUNM CLPQP MB**RR**N BPVIÑ MTIBV VEÑID ANSJA MTJOK MDODS ELPWI

UFOZM QMVNF O**HASE** SRJWR SFQCO TWVMB JGRPW **VSUE**X INQRS JEUEM

GGRBD GNNIL AGSJI DS**VSU E**EINT GRUEE TF**GGM P**ORDF OGTSS TOSEQ

OÑTG**R R**YVLP WJIFW XOTGG RPQ**RR** JSKET XRNBL ZETGG NEMUO TXJAT

ORVJH RSFHV NUEJI BC**HAS E**HEUE UOTIE FFGYA T**GGMP** IKTBW UEÑEN

IEEU.

En el criptograma se han subrayado y puesto en negrita algunas cadenas que se repiten. Las cadenas largas y más importantes son:

3 cadenas GGMP: separadas por 256 y 104 caracteres

2 cadenas YEDS: separadas por 72 caracteres

2 cadenas HASE: separadas por 156 caracteres

2 cadenas VSUE: separadas por 32 caracteres.

El valor del máximo común divisor de estas distancias debería ser un múltiplo de la longitud de la clave, esto es: mcd (256, 104, 72, 156, 32) = 4. Luego, la clave podría tener una longitud igual a cuatro caracteres. Hay que tener cuidado con elegir cadenas muy cortas ya que éstas pueden deberse solamente al azar y echar por tierra todas nuestras esperanzas de romper la cifra al aparecer una distancia cuyo valor sea por ejemplo primo relativo con las demás. En nuestro ejemplo sería el caso de elegir, entre otras, también la cadena VR (subrayada) que aparece tres veces al comienzo del criptograma con una separación de 65 y 31 caracteres, que no cumple con el máximo común divisor 4 encontrado anteriormente. Lo mismo ocurre con la cadena RR (subrayada), que se repite dos veces con una separación de 142 y luego 19 espacios. Para más inri, en ambos casos aparece un número primo, lo que asegura que el mcd sea igual a uno.

Si se sospecha que la clave tiene cuatro caracteres, se procede a romper el texto cifrado en cuatro criptogramas independientes C_A, C_B, C_C y C_D de 101 caracteres cada uno y que se llamarán subcriptogramas, tomando para el primero, segundo, tercero y cuarto los caracteres separados por cuatro espacios, siguiendo la escritura en columnas que se indicaba al comienzo de este apartado; es decir:

Primer subcriptograma: $C_1, C_5, C_9, ..., C_{391}, C_{401}$

Segundo subcriptograma: $C_2, C_6, C_{10}, ..., C_{398}, C_{402}$

Tercer subcriptograma: $C_3, C_7, C_{11}, ..., C_{399}, C_{403}$
Cuarto subcriptograma: $C_4, C_8, C_{12}, ..., C_{400}, C_{404}$

Por lo tanto:

C_A = PQAAEPDMRÑEEDCNUSRIECNIONSAAETLUOLAUIEULMNIIEAAOOLU MNARSOMRSISERNAISIRTMDTOORLIORRENENOAVSNIAEOFAMTEI

C_B = BVDÑTSBPPPDÑPPPBFDPQBUFNUEZCDFBÑMBEÑSFNPBBÑBÑNMKDPF QFSJFTBPUNJMBNGDUNUFPFSSÑRPFTPJTBTETTJFUBSUTFTPBÑE

C_C = VISSSIGSWWSDCQWZNMWVOEQMVIYESPHEEXEEEWMQRPMVISTMSWO MOEWQWJWEQEGDISSETEGOOSETYWWGQSXLGMXOHHECEEIGGIWEE

C_D = RCKDJEGLRYDRRMKVVTUVVDLWRKEYEHGSHVPLVHCPRVTVDJJDEIZ VHSRCVGVXRUGGLJVEGEGRGTQGVJXGRKRZGUJRRVJHHUEYGKUNU

Para descifrar los caracteres de la clave, suponiendo una cifra del tipo *Vigenère*, hay que encontrar el desplazamiento que se observa en cada uno de estos cifrados monoalfabéticos. Se aplicará un Método por Coincidencia Múltiple muy simple que se ha llamado Regla EAOS. Consiste en observar las frecuencias relativas de los caracteres de cada subcriptograma y marcar las cuatro mayores de forma que sigan modularmente la posición de las letras A, E, O y S en el alfabeto, las cuatro letras más frecuentes del lenguaje castellano. Evidentemente, si se desea una mayor precisión pueden tomarse más letras con el mismo sentido pero en la mayoría de los casos y para lo que aquí nos interesa es suficiente con estas cuatro. Por lo tanto, para estas cuatro letras, los caracteres m que las representen deberán tener una frecuencia relativa alta y estar situadas en las siguientes posiciones:

m mod 27 posición relativa de la letra A en el alfabeto desde el origen

m+4 mod 27 posición relativa de la letra E en el alfabeto desde el origen

m+15 mod 27 posición relativa de la letra O en el alfabeto desde el origen

m+19 mod 27 posición relativa de la letra S en el alfabeto desde el origen

Esto quiere decir que al ser todos los subcriptogramas resultado de cifrados monoalfabéticos, entonces alguna letra del texto cifrado tendrá aproximadamente la frecuencia característica de la letra A en el lenguaje, otra la de la letra E, otra de la letra O y otra de la letra S. Estos valores altos de frecuencia deberán estar separados por una relación de distancias constante pues de la A a la E hay cuatro espacios, de la E a la O hay once, de la O a la S cuatro y, por último, de la S a la A siete.

A continuación se puede ver una tabla con la contabilización de los caracteres de cada subcriptograma para este ejemplo.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
C_A	**11**	0	2	3	**12**	1	0	0	11	0	0	5	6	9	1	**10**	2	1	9	**7**	4	5	1	0	0	0	0
C_B	0	**14**	1	6	4	**12**	1	0	0	4	1	0	3	6	8	6	**14**	2	1	6	**9**	7	1	0	0	0	1
C_C	0	0	1	2	**18**	0	7	3	**7**	1	0	1	7	1	0	0	2	6	1	**12**	3	0	3	**12**	3	2	1
C_D	0	0	3	5	7	0	**12**	6	1	7	**5**	4	1	1	0	6	2	1	**13**	2	3	7	**14**	0	2	3	2

Tabla 4. Frecuencias de los monogramas del ejemplo de criptoanálisis.

De la tabla anterior, se tomarán los valores que se subrayan y están en negrita como caracteres en los cuales se cumple en buena medida la regla indicada, es decir:

$(m_A + 4) \bmod 27 = m_E$

$(m_E + 11) \bmod 27 = m_O$

$(m_O + 4) \bmod 27 = m_S$

$(m_S + 7) \bmod 27 = m_A$

Siendo m_A, m_E, m_O y m_S las posiciones de los caracteres con mayor frecuencia relativa en el criptograma que cumplan este sentido modular.

En este caso, para el criptograma C_A se observa que la única solución que cumple con la modularidad es aquella en la que el texto cifrado coincide con el texto en claro AEOS con valores (11, 12, 10, 7), luego es posible que la primera letra de la clave sea la A. El valor alto que muestra la letra I no cumple esta condición y se descarta. Para C_B la relación de letras con alta frecuencia en este orden modular está muy claro que se encuentra desplazada un carácter a la derecha, BFPT (14, 12, 14, 9) con lo cual la clave puede ser la letra B. Para el tercer criptograma C_C los números 18, 7, 12, y 12 cumplen con la modularidad exigida por la ecuación obteniendo ahora un ciclo EISW por lo que puede suponerse que la tercera letra de la clave es la E. Por último para el criptograma C_D, se eligen los caracteres RVGK con frecuencias (13, 14, 12, 5) con lo que la clave será la letra R. Con esto se llega a la conclusión de que la clave puede ser K = ABER. Se descifrará entonces el criptograma C según *Vigenère* con la clave ABER, «a ver» qué sucede.

```
C = PBVRQ VICAD SKAÑS DETSJ PSIED BGGMP SLRPW RÑPWY
K = ABERA BERAB ERABE RABER ABERA BERAB ERABE RABER
M = PARAQ UELAC OSANO MESOR PREND ACOMO OTROS AÑOSH
```

Como el texto «Para que la cosa no me sorprenda como otros años...» tiene sentido en castellano, y es imposible que se dé por pura casualidad, se sigue descifrando y obteniendo el texto que se indica.

«Para que la cosa no me sorprenda como otros años, he comenzado ya con unos suaves ejercicios de precalentamiento; mientras desayunaba he contemplado una bola plateada y una tira de espumillón y mañana me iniciaré en el amor al prójimo con los que limpien el parabrisas en los semáforos. Esta gimnasia del corazón metafórico es tan importante como la del otro corazón porque los riesgos coronarios están ahí, escondidos tras la vida sedentaria y parapetados en fechas como estas de Navidad.» *Comienzo del artículo "Gimnasia" del periodista Andrés Aberasturi publicado en el periódico El Mundo el 4/12/94.*

Observe que la cadena repetida GGMP de criptograma corresponde a la palabra "como", bastante común en nuestro lenguaje.

Si los sub-criptogramas son relativamente pequeños, a veces no resulta tan fácil decidir cuál es la clave mediante la observación de las frecuencias puesto que será muy aventurado hacer estadísticas con tan pocos datos. En este ejemplo con más de 100 caracteres por cada sub-criptograma, las estadísticas no funcionan tan mal como se ha visto. Por lo demás, conocido el origen del mensaje, la clave era obvia ¿verdad?

Ejemplo: Encuentre la longitud de la clave por el método de *Kasiski* a partir del criptograma obtenido con el algoritmo de *Vigenère*.

C = AWHFW ZAPIE ARXKS CXWEX WEXWE **KJPUR EBWTU S**COOB JGT**KJ PUREB WTUSC** NGXW.

Solución: Se observan tres repeticiones XWE seguidas, separadas entre sí por 3 espacios. Además, hay una larga cadena KJPUREBWTUSC que aparece dos veces, separada por 18 espacios. Como esto es muy "sospechoso", se calcula el mcd (3,18) = 3; luego este valor puede ser la longitud de la clave. De hecho la clave en esta cifra es la cadena PIO. Rompa la cifra en 3 subcriptogramas e intente comprobar si efectivamente se cumplen las estadísticas de frecuencia.

En el ejemplo anterior, si la clave que se utiliza es POLLOS en vez de PIO, se obtiene el criptograma C = ADDAD DAWEZ HVXQO XEAEE SZEAE **QFLBV EISOB WC**VLW PKT**QF LBVEI SOBW**C TCSD. En él se observa la cadena de caracteres EAE separada por 6 espacios, además de la cadena QFLBVEISOBWC separada por 18 espacios por lo que la longitud de la clave podría ser mcd (6,18) = 6. El descifrado con las dos claves y el posterior descubrimiento de esta canción de infancia se deja como ejercicio.

Debe tenerse en cuenta que este método al ser estadístico no es ni mucho menos infalible en el sentido que, por puro azar, se pueden dar cadenas repetidas en el criptograma cuya separación no sea múltiplo de la clave o, incluso peor, que sea un número primo como ya se ha comentado con lo que el máximo común divisor será igual a la unidad. En la práctica, para evitar estas soluciones falsas, deberá contarse con un criptograma de muchos caracteres, por ejemplo varias centenas, que incluso una vez roto en subcriptogramas se tenga cerca de la centena de caracteres en cada uno y luego buscar cadenas de caracteres largas, de una longitud mayor o igual a 3 y en lo posible que se repitan más de una vez.

Índice de Coincidencia

Otra forma de encontrar el período de un cifrador polialfabético es el conocido como Índice de Coincidencia, propuesto por *William F. Friedman* en una publicación de *Riverbank* en 1922. La publicación de "El Índice de Coincidencia y sus Aplicaciones en Criptografía" considerada durante muchos años como la mayor contribución al desarrollo de la criptología, entronca definitivamente las matemáticas y estadísticas con la criptología y permitió, entre otras cosas, criptoanalizar las máquinas de rotores con alfabetos progresivos que traían de cabeza a los criptólogos y servicios de inteligencia durante la Segunda Guerra Mundial.

En lo que aquí nos interesa, el Índice de Coincidencia medirá la variación o varianza de la frecuencia de aparición de los caracteres de un criptograma. La idea es la siguiente: en un sistema por sustitución simple monoalfabeto, con período igual a uno, se encuentra una importante variación en la frecuencia relativa de aparición de las letras. En cambio, para los sistemas polialfabéticos con un período grande, la variación de estas frecuencias es muy baja debido al efecto de difusión. Entonces en el primer caso existe un Índice de Coincidencia IC alto, y en el segundo este IC será bajo. Se definirá primeramente MD como la Medida de la Dispersión que nos entrega la variación de frecuencias en cada carácter, relativa a una distribución uniforme:

$$MD = \sum_{i=0}^{n-1} (p_i - \frac{1}{n})^2$$

donde p_i es la probabilidad de que un carácter cualquiera elegido aleatoriamente del criptograma sea el carácter i-ésimo a_i de un alfabeto con longitud n. Además, dado que:

$$\sum_{i=0}^{n-1} p_i = 1$$

Si n = 27, alfabeto castellano en mayúsculas, se tiene que:

$$MD = \sum_{i=0}^{26} p_i^2 - 0{,}037$$

La Medida de Dispersión MD evalúa la altura de los picos y los valles en una distribución de frecuencias con respecto a una distribución uniforme. Para el lenguaje castellano con n = 27, lógicamente 1/n = 0,037 será la línea base, de forma que los picos serán frecuencias relativas por sobre esta línea y los valles frecuencias relativas por debajo de la misma. Luego, si fr_M es la frecuencia relativa de la letra M, fr_M - 0,037 será el tamaño del pico o del valle observado y p_M - 0,037 el tamaño esperado del pico o del valle. Puesto que los picos serán positivos y los valles negativos, para que estos valores no se cancelen en la ecuación de la MD se utiliza $(p_i - 1/n)^2$.

Para una distribución uniforme, esto es las 27 letras del alfabeto equiprobables, se tiene MD = $27(1/27)^2 - 1/27 = 0$ que es lo esperado pues el Índice de Coincidencia indica la variación de la frecuencia de las letras respecto a una distribución uniforme. Es lógico que un texto que tenga una distribución de caracteres equiprobables presente una medida de dispersión igual a cero. En el otro extremo, si los caracteres del texto presentan la distribución característica del lenguaje castellano se tendrá:

$$\sum_{i=a}^{i=z} p_i^2 = p_a^2 + p_b^2 + p_c^2 + \ldots + p_x^2 + p_y^2 + p_z^2 = 0{,}072$$

Luego, la varianza será MD = 0,072 - 0,037 = 0,035. Esto quiere decir que la varianza de los caracteres de un criptograma tendrá un valor máximo igual a 0,035 cuando el cifrado haya sido monoalfabético y tiende a cero cuando el cifrado es polialfabético y el número de alfabetos utilizados es muy grande.

$$0 \leq MD \leq 0{,}035$$

El valor de p_i^2 significa la probabilidad de que al tomar dos caracteres aleatorios del criptograma, los dos sean iguales. Este valor se define como Índice de Coincidencia:

$$IC = \sum_{i=0}^{n-1} p_i^2$$

Como no se conoce el período ni las probabilidades p_i del criptograma, no será posible encontrar la Medida de la Dispersión, por lo menos de forma teórica. No se preocupe por esto ya que sí será

posible, no obstante, estimar MD usando la distribución de frecuencias de las letras del texto cifrado y aproximar así la probabilidad con la frecuencia observada. Luego, si f_i son las ocurrencias del carácter i en un criptograma de N letras, la probabilidad de elegir simultáneamente dos caracteres iguales de forma aleatoria, es decir p_i^2, será:

$$p_i^2 = \frac{f_i(f_i - 1)/2}{N(N-1)/2} = \frac{f_i(f_i - 1)}{N(N-1)}$$

Este resultado se explicará en el siguiente ejemplo. Luego IC será igual a:

$$IC = \frac{\sum_{i=0}^{n-1} f_i(f_i - 1)}{N(N-1)}$$

Ejemplo: Al cifrar M = ANIMAL RARO con método de *Beaufort* y K = CERDO se obtiene C = CRKRO RNRMA.

Encuentre la probabilidad de elegir dos caracteres iguales R en C.

Solución: C = CRKRO RNRMA. Como hay 10 letras en el criptograma y existen 4 caracteres R, el valor de f_R es igual a 4, a lo que se podría asociar una probabilidad $p_R = 0.4$

Se puede entonces concluir que el Índice de Coincidencia es un método para encontrar de forma aproximada la varianza que presentan los caracteres de un criptograma por medio de la observación de los datos. Pudiéndose deducir:

$$IC = MD + 0,037$$

En la expresión anterior el valor de IC puede ser calculado a partir de los valores encontrados en un criptograma ya que MD no es posible calcularlo como ya se había comentado en un párrafo anterior. No obstante, como el valor de MD se encuentra entre 0 y 0,035 se llega a la conclusión de que el Índice de Coincidencia IC estará comprendido entre los siguientes valores:

$$0,037 \leq IC \leq 0,072$$

Para cifradores con período d, el valor esperado del IC para un texto de N caracteres vendrá dado por la siguiente ecuación:

$$IC = (1/d)[(N-d)/(n-1)] *0,072 + [(d-1)/d](N/N-1) *0,037$$

d	IC	d	IC
1	0,072	5	0,044
2	0,054	10	0,040
3	0,049	...	...
4	0,046	Grande	0,037

Tabla 5. Índice de coincidencia para cifras con período *d*

El valor de IC = 0,072 para un período d igual a la unidad, esto es un cifrador por desplazamiento puro monoalfabético, nos indica que dicho texto será equivalente al lenguaje castellano en lo que a distribución de frecuencias relativas de los caracteres se refiere. Luego, todo cifrado monoalfabético tendrá este valor que coincidirá con el de un texto en claro, en tanto que aquí no interesa que sea precisamente la letra E la que presente una frecuencia de aparición del 13%; puede ser cualquier otro carácter del criptograma, dependiendo del desplazamiento utilizado en el cifrado.

El Índice de Coincidencia IC presenta una fuerte variación para valores pequeños del período de cifrado, no así para valores grandes. Por este motivo, en el criptoanálisis de sistemas por sustitución, el método se usa conjuntamente con el de *Kasiski* puesto que, si bien no es preciso en cuanto al número de alfabetos utilizados, sí lo es para indicar si se trata de una sustitución monoalfabeto o polialfabeto.

Ejemplo: Utilizando la ecuación del Índice de Coincidencia, determinar si el siguiente criptograma pertenece a un cifrado por sustitución monoalfabética o polialfabética.

> C = WVKNK BCOFQ NCGEW CEKQO FGNKO FKEGF GEQKO EKFGO EKCFG VGTÑK OCTUK GNUKI WKGOV GETKR VQITC ÑCRGT VGOGE GCWOE KHTCF QRQTU WUVKV WEKQO ÑQOQC NHCDG VKECQ RQNKC NHCDG VKEC

Solución: Los 139 caracteres del criptograma y su frecuencia son:

> A=0, B=1, C=15, D=2, E=12, F=7, G=16, H=3, I=2, J=0, K=19, L=0, M=0, N=7, Ñ=3, O=11, P=0, Q=11, R=4, S=0, T=7, U=4, V=9, W=6, X=0, Y=0, Z=0. Se obtiene: IC = 0,071. Como este valor se aproxima mucho a 0,072 se concluye que se trata de un cifrado de tipo monoalfabético. ¿Cuál sería el mensaje?

No siempre será posible encontrar el período usando el Índice de Coincidencia. Cuando la clave es relativamente larga, más de 4 caracteres, será mucho más confiable el método de *Kasiski*. De todas maneras, el Índice de Coincidencia nos permitirá determinar, una vez fraccionado el criptograma en sub-criptogramas por *Kasiski*, si cada uno de ellos se trata de un cifrado monoalfabético, comparando el valor encontrado del IC en cada uno de ellos con el característico del lenguaje castellano.

En el ejemplo del artículo "Gimnasia" del ataque por *Kasiski* visto anteriormente, para cada uno de los cuatro sub-criptogramas se tienen los siguientes valores de IC:

$$C_A \Rightarrow IC = 0{,}070$$
$$C_B \Rightarrow IC = 0{,}073$$
$$C_C \Rightarrow IC = 0{,}075$$
$$C_D \Rightarrow IC = 0{,}065$$

Como todos los valores se acercan bastante al IC característico del lenguaje castellano (IC = 0,072), puede asegurarse que efectivamente cada uno de los cuatro criptogramas hallados se trata de un cifrado monoalfabético como era el caso. En resumen, si se desea atacar un criptograma que se supone se ha obtenido mediante sustitución con más de un alfabeto, habrá que usar las siguientes herramientas en este orden:

a) Análisis de la distribución de frecuencias del texto del criptograma. Si es parecida a la del lenguaje, el cifrado será monoalfabético; caso contrario, será polialfabético.

b) Cálculo del Índice de Coincidencia IC para confirmar que el criptograma se debe a un cifrado polialfabético y tener una primera idea del período del cifrador.

c) Aplicación del método de *Kasiski* para encontrar el período, obteniendo varios subcriptograma. Cálculo luego del IC de cada uno de dichos sub-criptogramas para asegurarse que se trata de un cifrado con un desplazamiento constante.

d) Uso del Método de Coincidencia Múltiple o Regla EAOS para encontrar las letras que forman la clave.

e) Encontrada la clave, se procede a descifrar el criptograma siempre en el supuesto de que es conocido el algoritmo de cifra.

2.4.3.3 Cifradores polialfabéticos no periódicos

La debilidad de los cifradores por sustitución con más de un alfabeto está en la periodicidad de la clave. Esto provoca posibles cadenas repetidas en el criptograma que entrega una pista al criptoanalista, facilitando sobremanera el ataque a estos cifrados. Como la fortaleza del cifrado o distancia de unicidad dependía de la longitud de la clave o período, la solución consistirá en aumentar la longitud de esta clave.

¿Qué sucede si se aumenta la longitud de la clave de forma que tenga un tamaño igual o mayor que el texto en claro? Esto sería lo mismo que adoptar el criterio propuesto por *Shannon* para un sistema con secreto perfecto. En este caso los criptogramas soportarían el ataque por el método de *Kasiski* puesto que al no haber período alguno, sería imposible dividir el criptograma en otros menores.

Cifrador con clave continua

Si se acepta que la fortaleza de un cifrado por sustitución utilizando una clave tan larga como el mensaje se verá aumentada, el problema está ahora en determinar cómo se genera una clave con tales características. Una solución sencilla podría ser elegir como clave un texto, conocido por el transmisor y el receptor claro está, con una cantidad de caracteres a lo mínimo igual que la del mensaje en claro.

Por lo tanto, ya no se habla de una clave sino de una secuencia de clave y el cifrador en cuestión dejará de cifrar bloques con una clave periódica para convertirse en un cifrador de flujo propiamente dicho. Si el método de cifrado es similar al de *Vigenère*, este criptosistema se conoce como cifrador con clave continua.

Ejemplo: Cifre el mensaje que se indica con el algoritmo de *Vigenère*, utilizando como secuencia de clave el párrafo primero del libro "Cien años de soledad".

M = INFORMAMOS NEGATIVAMENTE LA COMPRA DE ACCIONES

Solución: Se escribe el mensaje y la clave conjuntamente y se cifra cada par M_i/k_i con la tabla de *Vigenère*.

```
M = INFORMAMOSNEGATIVAMENTELACOMPRADEACCIONES
K = MUCHOSAÑOSDESPUESFRENTEALPELOTONDEFUSILAM
C = THHVGEAZDLPIYPÑMÑFDIZNILLRSWELOPHEHWAWXEE
```

La operación de descifrado de los cifradores de clave continua es obvia. Basta aplicar la operación inversa del desplazamiento i-ésimo al igual que en los cifradores polialfabéticos, que para el caso de *Vigenère* era $M_i = (C_i - k_i) \bmod n$.

Ejemplo: Descifre el criptograma cifrado con *Vigenère* conociendo que la secuencia de la clave es el texto de García Lorca que se indica.

C = GSJFE OPEEO UCUGC EIWGP OVWUR WXPPN MRZOL HEMJO PSIEY PLUGZ LWHCS GETNL C.

K = VERDE QUE TE QUIERO VERDE, VERDE VIENTO, VERDES RAMAS. EL BARCO SOBRE LA MAR Y EL CABALLO EN LA MONTAÑA.

Solución: Realizando la operación de descifrado entre C_i y k_i se obtiene:

C = GSJFE OPEEO UCUGC EIWGP OVWUR WXPPN MRZOL HEMJO PSIEY PLUGZ LWHCS GETNL C

K = VERDE QUETE QUIER OVERD EVERD EVIEN TOVER DESRA MASEL BARCO SOBRE LAMAR Y

M = LORCA YVALL EINCL ANSON LASDO SCIMA SDELT EATRO ESPAÑ OLDEL SIGLO VEINT E

El mensaje original es por lo tanto:
M = Lorca y Valle Inclán son las dos cimas del teatro español del siglo veinte.

2.4.3.4 Criptoanálisis de los cifrados con clave continua

A pesar de que el espacio de claves es tan grande o más que el de los mensajes, los cifradores con clave continua vistos en el apartado anterior no nos entregan el ansiado secreto perfecto; paciencia, ya llegará un sistema con tales características. La razón es que tanto el texto en claro como el texto de la clave presentarán la redundancia característica del lenguaje castellano.

William Friedman propone un método que lleva su nombre y que consiste, básicamente, en observar que una importante cantidad de pares de letras del mensaje en claro y de la secuencia de clave caen dentro de lo que se han considerado monogramas de alta frecuencia del lenguaje. Esto es, existirán pares M_ik_i en los que tanto el carácter M_i del texto en claro como el carácter k_i de la clave tienen ambos una alta frecuencia. Esto reducirá muchísimo el trabajo del criptoanálisis, si se compara con la fuerza bruta de hacer coincidir a cada letra del criptograma todas y cada una de las letras del alfabeto como posibles claves, lo que significaría por ejemplo para los cinco primeros caracteres del criptograma evaluar 27^5 = 14.348.907 emparejamientos.

Friedman recomienda suponer inicialmente que todos los caracteres del criptograma se corresponden con pares M_ik_i de alta frecuencia. Luego, para cada letra C_i del texto cifrado se escribe como texto en claro el propio alfabeto y como letra clave aquella que da origen al elemento C_i en cuestión. Como se verá a continuación, la distribución de la clave sobre el alfabeto en claro seguirá la fórmula del cifrador de *Beaufort*. Hecho esto, se procede a buscar los pares M_i y k_i de alta frecuencia que permitirán ir descubriendo simultáneamente el texto en claro y la secuencia de clave. Por simplicidad, suponga que los caracteres de alta frecuencia se corresponden con los nueve que forman la palabra

ESTIRANDO. ¿Cómo funciona el método con un texto elegido a propósito para que las cosas nos salgan bien a la primera? Se verá en el siguiente ejemplo.

M = SI ESTÁS CANSADO Y HARTO DE TODO, DESCANSA

K = ANTES DE TIRARTE AL RÍO, PIÉNSALO DOS VECES

Si se cifra el texto con dicha clave, y marcamos en negritas los pares M/K de alta frecuencia se obtiene:

M = **SIEST** **AS**C**AN** **SADO**Y H**ART**O **DETOD** O**DES**C A**N**S**A**
K = **ANTES** **DE**T**IR** **ARTE**A L**RIO**P **IENSA** L**ODO**S V**E**C**E**S
C = SUXWM DWVIE SRWSY RRZIE LIGHD ZRHHU VQUE

Observe que, debido al tipo de mensaje y clave elegidos, casi todos los pares de letras de TextoEnClaro y TextoCifrado que generan C son de alta frecuencia. Están marcados en negrita y se indican a continuación.

C_i	M_iK_i	C_i	M_iK_i	C_i	M_iK_i	C_i	M_iK_i
S	SA	W	SE	Z	RI	D	DA
U	IN	I	AI	I	TO	R	DO
X	ET	E	NR	L	DI	H	ED
W	SE	R	AR	I	EE	H	SO
M	TS	W	DT	G	TN	Q	NE
D	AD	S	OE	H	OS	E	AE

Precisamente esto es lo que nos permitirá romper la cifra. Existirán 27 posibles emparejamientos de letras del texto en claro/clave para cada letra del criptograma. Se consideran las cinco primeras letras del criptograma anterior (SUXWM), ordenan los alfabetos de mensaje y clave de forma que se obtenga siempre S, U, X, W y M. Luego se tomarán los pares de alta frecuencia: observe lo que sucede.

Letra en claro:	**A**	B	C	D	**E**	F	G	H	I	J	K	L	M	N	Ñ	**O**	P	Q	R	**S**	T	U	V	W	X	Y	Z
Letra clave:	**S**	R	Q	P	**O**	Ñ	N	M	L	K	J	I	H	G	F	**E**	D	C	B	**A**	Z	Y	X	W	V	U	T
Letra criptograma:	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S

Letra en claro:	A	B	C	**D**	E	F	G	H	**I**	J	K	L	M	**N**	Ñ	O	P	Q	**R**	S	T	U	V	W	X	Y	Z
Letra clave:	U	T	S	**R**	Q	P	O	Ñ	**N**	M	L	K	J	**I**	H	G	F	E	**D**	C	B	A	Z	Y	X	W	V
Letra criptograma:	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U

Letra en claro:	A	B	C	D	**E**	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	**T**	U	V	W	X	Y	Z
Letra clave:	X	W	V	U	**T**	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	**E**	D	C	B	A	Z	Y
Letra criptograma:	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Letra en claro:	A	B	C	**D**	**E**	F	G	H	**I**	J	K	L	M	N	Ñ	**O**	P	Q	R	**S**	**T**	U	V	W	X	Y	Z
Letra clave:	W	V	U	**T**	**S**	R	Q	P	**O**	Ñ	N	M	L	K	J	**I**	H	G	F	**E**	**D**	C	B	A	Z	Y	X
Letra criptograma:	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Letra en claro:	A	B	C	D	**E**	F	G	H	**I**	J	K	L	M	N	Ñ	O	P	Q	R	**S**	**T**	U	V	W	X	Y	Z
Letra clave:	M	L	K	J	**I**	H	G	F	**E**	D	C	B	A	Z	Y	X	W	V	U	**T**	**S**	R	Q	P	O	Ñ	N
Letra criptograma:	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M

Criptograma:

S	U	X	W	M
A - S	D - R	**E - T**	D - T	E - I
E - O	**I - N**	T - E	E - S	I - E
O - E	N - I		I - O	S - T
S - A	R - D		O - I	**T - S**
			S - E	
			T - D	

Los anteriores son pares de alta frecuencia LetraEnClaro - LetraDeClave para estas cinco primeras letras del criptograma. Existirán 4 x 4 x 2 x 6 x 4 = 768 posibles pentagramas de texto en claro con secuencia de clave, mucho menos que los 14 millones anteriores. Uno de ellos será el texto en claro SIEST con la clave ANTES, precisamente los pares 4º (S-A), 2º (I-N), 1º (E-T), 5º (S-E) y 4º (T-S) que se han macado en negrita.

Mientras menor sea esta ventana habrá menor probabilidad de hallar la letra verdadera pero, por otro lado, si se consideran todos los caracteres, el número de combinaciones puede volverse intratable. Con los nueve caracteres de la palabra ESTIRANDO se obtienen para cada letra del alfabeto al menos un par M_iK_i de alta frecuencia, excepto cuando el elemento del criptograma es la letra Y. A continuación se verá un ejemplo en el que no resulta afortunada la elección del bloque para comenzar el ataque a un cifrado continuo. Sean el mensaje M y la clave K:

M = SE SUPONE QUE APARECEN DIGRAMAS FRECUENTES EN LOS CIFRADOS
K = CUANDO USAMOS COMO CLAVE UN TEXTO QUE TAMBIÉN ES REDUNDANTE

M = SE**S**UP **ONE**QU **EA**PAR **E**CE**N**D **I**G**RA**M A**S**FRE CU**E**NT **ESEN**L **OS**CIF **RADOS**
K = CU**A**ND **OUS**AM **OS**COM **O**CL**A**V **E**U**NT**E X**T**OQU ET**A**MB **IENE**S **RE**DUN **DANTE**
C = UYSHS DHWQG SSROD SEONY MAETP XMTIY GÑEYU MWQQD GWFCR UAPIW

Si se toman las tres primeras letras del criptograma anterior (UYS) y buscan los pares de alta frecuencia, se obtiene ahora:

Letra en claro:	A	B	C	**D**	E	F	G	H	**I**	J	K	L	M	**N**	Ñ	O	P	Q	**R**	S	T	U	V	W	X	Y	Z
Letra clave:	U	T	S	**R**	Q	P	O	Ñ	**N**	M	L	K	J	**I**	H	G	F	E	**D**	C	B	A	Z	Y	X	W	V
Letra criptograma:	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U

Letra en claro:	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Letra clave:	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z
Letra criptograma:	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Letra en claro:	**A**	B	C	D	**E**	F	G	H	I	J	K	L	M	N	Ñ	**O**	P	Q	R	**S**	T	U	V	W	X	Y	Z
Letra clave:	**S**	R	Q	P	**O**	Ñ	N	M	L	K	J	I	H	G	F	**E**	D	C	B	**A**	Z	Y	X	W	V	U	T
Letra criptograma:	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S

Los pares de alta frecuencia LetraTextoClaro - LetraSecuenciaClave serán:

Criptograma

```
 U       Y       S
D - R    -     A - S
I - N    -     E - O
N - I    -     O - E
R - D    -     S - A
```

En este caso la elección del comienzo del criptograma en donde aparece la letra Y que es la única que no presenta pares de alta frecuencia con ESTIRANDO, nos ha llevado a un callejón sin salida. No hay de qué preocuparse, en este tipo de cifra nos será de mucha utilidad la redundancia del lenguaje. Al elegir los 9 caracteres de la palabra ESTIRANDO como los de alta frecuencia del lenguaje, se cumple esta condición en pares M_ik_i para aproximadamente el 50% del texto por lo que con un poco de imaginación podrá llegar a descifrar el mensaje.

Puede entonces concluirse que si se desea cifrar un texto en claro mediante un cifrador con clave continua y que éste tenga un secreto perfecto, deberá elegir una secuencia de clave aleatoria como sería, por ejemplo, un listado de nombres o direcciones de una guía de teléfonos a partir de una página determinada; si esa guía es antigua y más aún de otro país, tanto mejor. La segunda condición que debería cumplir el sistema es que la clave sea única por lo que, tras ser utilizada, debería destruirse. Este principio da lugar a los denominados criptosistemas de uso o control único también denominados genéricamente one-time pad, comunes hoy en día.

Cifrador de Vernam

En 1917 *Gilbert S. Vernam*, nativo de Brooklyn e ingeniero del MIT que trabaja en los laboratorios de la empresa AT&T, diseña un dispositivo criptográfico para comunicaciones telegráficas basado en los 32 códigos *Baudot* de los teletipos desarrollados por su compañía. *Vernam* propone el uso de una clave continua con la que cifrar el mensaje en claro carácter por carácter para obtener el texto cifrado. Posteriormente, junto a *Joseph Mauborgne* (capitán US *Army Signal Corps*) observaron la dificultad de criptoanalizar comunicaciones si la clave era aleatoria y se cumplían una serie de propiedades. Esto dio lugar al nacimiento de la libreta de un solo uso. En la década de los 40, *Claude Shannon* demostró, usando elementos de la teoría de la información, que este sistema tenía la propiedad que él llamó secreto perfecto, esto es, el texto cifrado no proporciona absolutamente ninguna información acerca del texto en claro. Por tanto, la probabilidad *a priori* de un mensaje en claro *M* es igual que la probabilidad *a posteriori* de un mensaje en claro *M* dado el correspondiente texto cifrado. Y de hecho todos los textos en claro son igualmente probables. El cifrado de *Vernam* (libreta de un solo uso) es el único algoritmo criptográfico conocido irrompible si se dan una serie de condiciones: libretas de un solo uso perfectamente aleatorias, la generación e intercambio de las libretas de un solo uso tiene que ser segura, y la libreta tiene que ser al menos tan larga como el mensaje, y hace falta un tratamiento

cuidadoso para asegurarse de que siempre permanecerán en secreto para cualquier adversario. Es necesario deshacerse de ellas correctamente para evitar cualquier reutilización parcial o completa.

En la práctica, el gran problema de este algoritmo es conseguir que la clave cumpla las condiciones indicadas. A lo largo de la historia se han documentado múltiples casos donde este algoritmo ha sido vulnerado por hacer caso omiso, o no poner las precauciones adecuadas, en alcanzar las características indicadas. Un ejemplo curioso fue el proyecto VENONA.

Independientemente de estas cuestiones, en esencia el cifrador de *Vernam* representa cada carácter M_i con 5 bits en código *Baudot* que se suma or-exclusivo (módulo 2) con la correspondiente clave k_i de una secuencia binaria aleatoria. De esta forma, el cifrador de *Vernam* genera un flujo de bits de texto cifrado de la forma:

$$C = E_K(M) = C_1C_2C_3 \ldots C_N$$

donde:

$$C_i = (M_i + k_i) \bmod 2 \text{ para } i = 1,2,\ldots,N$$

$$C_i = M_i \oplus k_i$$

Para la operación de descifrado, se utiliza el mismo algoritmo por la propiedad involutiva de la operación or-exclusivo. Esto es:

$$C_i \oplus k_i = (M_i \oplus k_i) \oplus k_i$$

Como $k_i \oplus k_i = 0$ para $k_i = 0$ y $k_i = 1$, se obtiene:

$$C_i \oplus k_i = M_i$$

En la siguiente figura se muestra un cifrador de *Vernam* binario como el descrito.

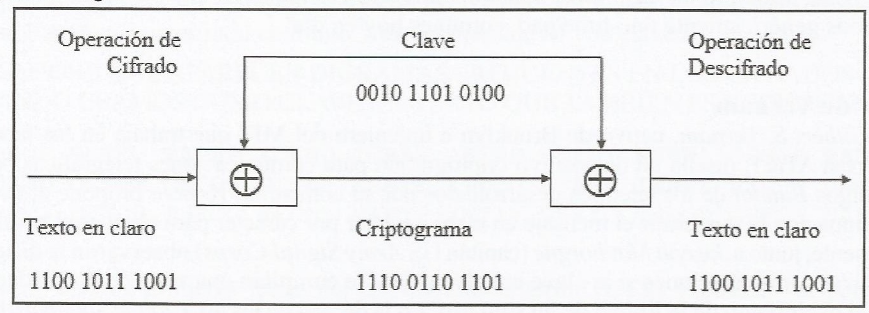


Figura 6. Esquema de un cifrador de *Vernam* binario.

Ejemplo: Usando el código de *Baudot*, cifre el mensaje M = BYTES con la clave K = VERNAM.

Solución: Haciendo la suma OR exclusivo C=UHGFI

$B \oplus V = 11001 \oplus 11110 = 00111 = U$
$Y \oplus E = 10101 \oplus 00001 = 10100 = H$
$T \oplus R = 10000 \oplus 01010 = 11010 = G$
$E \oplus N = 00001 \oplus 01100 = 01101 = F$
$S \oplus A = 00101 \oplus 00011 = 00110 = I$

Para descifrar, como ya se ha dicho, sencillamente se aplica nuevamente la operación del or-exclusivo como se indica en el siguiente ejemplo.

Ejemplo: Se recibe el siguiente criptograma C = 00110 10100 11100 11010 00000 00010 01110 01011 00110 de un texto con código *Baudot* que se ha cifrado con clave la K = Gloria Estefan. Descífrelo.

Solución: Se busca el valor binario del código *Baudot* y con la operación suma OR exclusivo entre C y K ($C \oplus K$) se obtiene:

K=G: $00110 \oplus 11010 = 11100$ = M
K=L: $10100 \oplus 10010 = 00110$ = I
K=O: $11100 \oplus 11000 = 00100$ =
K=R: $11010 \oplus 01010 = 10000$ = T
K=I: $00000 \oplus 00110 = 00110$ = I
K=A: $00010 \oplus 00011 = 00001$ = E
K=_: $01110 \oplus 00100 = 01010$ = R
K=E: $01011 \oplus 00001 = 01010$ = R
K=S: $00110 \oplus 00101 = 00011$ = A
Luego M = MI TIERRA.

Como un divertimento más, puede representarse un cifrador de *Vernam* orientado a caracteres. En este caso la operación de cifra se realiza a través de desplazamientos módulo 27, como si se tratase de un cifrador monoalfabético, con una secuencia de clave compuesta por números aleatorios NA = k_i, como se indica.

M	=	C	I	F	R	A	D	O	R	D	E	V	E	R	N	A	M
M_i	=	2	8	5	18	0	3	15	18	3	4	22	4	18	13	0	12
k_i	=	73	12	39	81	07	28	95	52	30	18	32	29	47	20	07	62
M_i+k_i	=	75	20	44	99	7	31	110	70	33	22	54	33	65	33	7	74
C	=	U	T	Q	R	H	E	C	P	G	V	A	G	L	G	H	T

Los valores utilizados para la secuencia de clave en el ejemplo anterior están comprendidos entre 00 y 99, si bien pueden reducirse mod 27 y, por tanto, trabajar sólo con el CCR(27). Del ejemplo anterior hay que destacar un aspecto interesante de un cifrador de *Vernam*: en el criptograma aparecen caracteres iguales que provienen del cifrado de caracteres distintos del texto en claro, al igual que en todos los sistema polialfabéticos, como es el caso de las letras D, E y N que se cifran como el elemento G. Ahora bien, además de utilizar los 27 posibles alfabetos dependiendo del valor de la clave, al ser ésta aleatoria y carecer de periodicidad alguna, hace imposible cualquier tipo de ataque conociendo únicamente el criptograma. Si la secuencia aleatoria de clave usada luego se destruye, entonces el secreto es perfecto. El problema que persiste en este esquema es la transmisión segura de la clave secreta; para ello habrá esperar hasta el año 1977 en que se presenta el sistema de cifra de clave pública y se soluciona con este método el problema del intercambio de clave.

2.4.4 Cifradores por sustitución poligrámica monoalfabeto

Los cifradores poligrámicos, a diferencia de los monográmicos que cifraban carácter a carácter, consideran un poligrama con $n \geq 2$ del texto en claro para proceder a su cifrado. De esta forma, el bloque de información a cifrar serán digramas, trigramas o, en general, poligramas. El objeto de este cifrado por sustitución es destruir la distribución de frecuencia típica de los monogramas que,

al hacer coincidir el carácter M_i con el elemento cifrado C_i, posibilita el ataque por inspección de frecuencias en el caso de los monoalfabéticos o bien el criptoanálisis mediante método de *Kasiski*, Índice de Coincidencia y el método de *Friedman* para los polialfabéticos.

Si el cifrador transforma los digramas M_iM_{i+1} del texto en claro en criptogramas C_iC_{i+1}, se tendrá que:

$$M = M_1M_2 \cup M_3M_4 \cup \ldots \cup M_{N-1}M_N$$

en donde el signo $\cup$ indica unión de caracteres y N es el número total de caracteres del mensaje. Luego:

$$E_K(M) = E_K(M_1M_2) \cup E_K(M_3M_4) \cup \ldots \cup E_K(M_{N-1}M_N)$$
$$E_K(M) = C_1C_2 \cup C_3C_4 \cup \ldots \cup C_{N-1}C_N$$

De los cifradores poligrámicos, los más conocidos son los de *Playfair* de mediados del siglo XIX, que hace uso de una tabla de cifrar similar a la de *Polybios*, y el de *Hill*, que data de comienzos del siglo XX y que tiene una importancia especial en la criptografía clásica por el hecho de utilizar la matemática de matrices en módulo n para las operaciones de cifrado y descifrado. A comienzos del año 1999, la posibilidad del uso de matrices en los sistemas de cifra volvió a ser considerado, tras la aparición de un algoritmo de clave pública propuesto por una joven irlandesa aunque no tenía nada que ver con el sistema de *Hill*. No obstante, al final se demostró que el tal invento no era tan espectacular como se suponía.

Antes de destacar los aspectos más notorios del cifrado *Playfair* y el de *Hill* es bueno recordar a modo de cultura general el cifrador de *Polybios*.

Cifrador de Polybios

A mediados del siglo II antes de J.C., aparece el cifrador por sustitución de caracteres más antiguo que se conoce. Atribuido al historiador griego *Polybios*, el sistema de cifra consistía en hacer corresponder a cada letra del alfabeto un par de letras que indicaban la fila y la columna en la cual aquella se encontraba, en un recuadro de 5 x 5 = 25 caracteres, transmitiéndose por tanto en este caso el mensaje como un criptograma. Se muestra a continuación una tabla de cifrar de *Polybios* adaptada al inglés, con un alfabeto de cifrado consistente en el conjunto de letras A, B, C, D y E, aunque algunos autores representan el alfabeto de cifrado también como los números 1, 2, 3, 4 y 5.

	A	**B**	**C**	**D**	**E**
A	A	B	C	D	E
B	F	G	H	IJ	K
C	L	M	N	O	P
D	Q	R	S	T	U
E	V	W	X	Y	Z

	1	**2**	**3**	**4**	**5**
1	A	B	C	D	E
2	F	G	H	IJ	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Tabla 6. Tablas de cifrar de *Polybios*.

Acorde con este método, la letra A se cifrará como AA, la H como BC, etc. Esto significa que se aplica una sustitución al alfabeto {A, B, C, .., X, Y, Z} de 26 letras convirtiéndolo en un alfabeto de cifrado {AA, AB, AC, ..., EC, ED, EE} de 25 caracteres, si bien sólo existen 5 símbolos diferentes

{A, B, C, D, E}. Este tipo de tabla o matriz de cifrado será muy parecida a la que en el siglo XIX se utilizará en el criptosistema conocido como cifrador de *Playfair* y que será tratado en el apartado de cifradores poligrámicos, salvo que en este último la operación de cifra no se realiza por monogramas como en el de *Polybios* sino por digramas, conjunto de dos caracteres del texto en claro.

Ejemplo: Usando la Tabla del cifrador de *Polybios*, cifre el mensaje:

M = QUE BUENA IDEA LA DEL GRIEGO

Solución: C = DADEAE ABDEAECCAA BDADAEAA CAAA ADAECA BBDBBDAEBBCD

Aunque resulte elemental, se deja como ejercicio para el lector encontrar el criptograma cuando se utiliza la tabla de *Polybios* con representación numérica. El criptograma que se obtiene con este cifrador tiene una extensión de caracteres igual al doble de la del texto en claro, característica que no puede considerarse precisamente como una virtud de este método de cifra. En realidad no fue tan buena la idea.

2.4.4.1 Cifrador de Playfair

El cifrador de *Playfair* en realidad fue inventado por *Charles Wheatstone* para comunicaciones telegráficas secretas en el año 1854. No obstante, se le atribuye a su amigo el científico *Lyon Playfair* quien lo presenta a las autoridades inglesas de la época. Nuevamente eso se llama ser un buen amigo. Utilizado por el Reino Unido en la Primera Guerra Mundial, este sistema consiste en separar el texto en claro en digramas y proceder a su cifra de acuerdo a una matriz alfabética de dimensiones 5x5 en la cual se encuentran representadas 25 de las 26 letras del alfabeto inglés.

Para que este método de cifra presente un mayor nivel de seguridad, se incluirá al comienzo de dicha matriz una clave que se escribe a partir de la primera fila omitiendo las letras repetidas. A continuación de dicha clave, se distribuyen las restantes letras del alfabeto hasta completar toda la matriz. Por ejemplo, si la clave es VERANO AZUL (memorable serie española de TV de *Antonio Mercero*), la matriz será:

En las tablas siguientes se muestran las matrices comentadas para el lenguaje castellano de 27 caracteres. En este caso puede suponerse que las letras I y J ocupan una única celda, al igual que la Ñ y la N. La segunda matriz lleva como clave VERANO AZUL.

A	B	C	D	E
F	G	H	I/J	K
L	M	N/Ñ	O	P
Q	R	S	T	U
V	W	X	Y	Z

Tabla 7. Matriz de cifra de *Playfair.*

V	E	R	A	N/Ñ
O	Z	U	L	B
C	D	F	G	H
I/J	K	M	P	Q
S	T	W	X	Y

Tabla 8. Matriz de cifrado de *Playfair* con clave VERANO AZUL.

El método de *Playfair* cifrará pares de caracteres del texto en claro M_1M_2 como C_1C_2 de acuerdo a las siguientes reglas:

a) Si M_1 y M_2 se encuentran en la misma fila, se eligen los elementos del criptograma C_1 y C_2 como aquellos que están a la derecha de M_1 y M_2, respectivamente. Esta operación se realiza módulo 5, de forma que para la matriz se cumplen las siguientes transformaciones:

Pares del texto en claro	Criptograma
EA (1ª fila)	RN
LU (2ª fila)	BL
DH (3ª fila)	FC

b) Si M_1 y M_2 se encuentran en la misma columna, se eligen C_1 y C_2 como los caracteres que están inmediatamente debajo de ellos, operando módulo 5. Para la matriz se cumplen las siguientes transformaciones:

Pares del texto en claro	Criptograma
ED (2ª columna)	ZK
FU (3ª columna)	MF
AX (4ª columna)	LA

c) Si M_1 y M_2 se encuentran en filas y columnas distintas, entonces forman dos vértices de un rectángulo. Los elementos C_1 y C_2 se obtienen de los dos vértices que faltan para completar dicha figura geométrica, considerando siempre la fila de M_1 como el elemento C_1. Esto es, en la matriz se cumplen las siguientes transformaciones:

Pares del texto en claro	Criptograma
OT	ZS
YU	WB

Recuerde, además, que las letras I y J ocupan una misma celda, al igual que la N y la Ñ, por lo que se cumplen por ejemplo también las siguientes transformaciones en dicha matriz:

Pares del texto en claro	Criptograma
MI = MJ	PK
EN = EÑ	RV

No obstante, si en la operación de descifrado se cae en la retícula I/J, siempre se descifrará como la letra I.

d) Al representar el texto en claro como una cadena de digramas, pueden aparecer caracteres repetidos con lo cual no podría aplicarse ninguna de las tres opciones de cifrado anteriores. Tal sería el caso de cifrar el siguiente mensaje tenebroso:

M = LAS SOMBRAS LLAMAN A LA PUERTA DEL CASTILLO

M = LA <u>SS</u> OM BR AS <u>LL</u> AM AN AL AP UE RT AD EL CA ST IL LO

La solución a este problema (los digramas segundo SS y sexto LL) está en romper esta repetición antes del cifrado, incluyendo una letra nula que, de acuerdo al lenguaje castellano podría ser X, Z o Q por ejemplo. Por último, se usará en el texto la letra X como carácter de relleno, con lo que el mensaje se transforma ahora en:

M = LA SX SO MB RA SL LA MA NA LA PU ER TA DE LC AS TI LX LO

Observe que al incluir este carácter de relleno, ha desaparecido la repetición LL del digrama sexto, si bien aparece otro nuevo (LL) ahora en la posición 18 que se rompe de igual manera añadiendo la letra X.

e) Por último, es posible que el mensaje final a cifrar, una vez eliminados los digramas repetidos, tenga un número impar de caracteres. En tal situación se añade un carácter nulo al final de la cadena para poder cifrar el último carácter del texto en claro. Por ejemplo, si al mensaje anterior se le añade la palabra HOY, además de ser más inquietante, quedaría como sigue:

M = LA SX SO MB RA SL LA MA NA LA PU ER TA DE LC AS TI LX LO HO YX

Ejemplo: Cifre el mensaje, M = «Las sombras llaman a la puerta del castillo hoy» con una matriz de *Playfair* con clave K = MIEDO.

Solución: Siguiendo la matriz con clave MIEDO, se cifran los digramas del mensaje, a saber:

M = LA SX SO MB RA SL LA MA NA LA PU ER TA DE LC AS TI LX LO HO YX, obteniendo:
C = HCXEUEIA QBXSHCAH HFHCUZIS QFODSLCQ RDSEPEPM ZY

La matriz de cifra y comprobación de la misma es tarea suya.

Para descifrar un criptograma obtenido mediante *Playfair*, simplemente puede utilizarse el algoritmo inverso, esto es:

a) Si los elementos C_1 y C_2 están en la misma fila, se eligen M_1 y M_2 como los caracteres inmediatamente a la izquierda, módulo 5.

b) Si los elementos C_1 y C_2 se encuentran en la misma columna, se eligen M_1 y M_2 como los caracteres inmediatamente arriba de aquellos, módulo 5.

c) Si los elementos C_1 y C_2 están en filas y columnas distintas, M_1 y M_2 se eligen como aquellos caracteres que forman los vértices que faltan del recuadro, comenzando por la fila del primer elemento C_1.

Ejemplo: Si en la matriz de cifra de *Playfair* con la clave BEATLES se eliminan los caracteres K y Ñ, con relleno X, descifre el siguiente criptograma:

C = EC TB AZ EN WB JH TX AB BU VC LO JT PM IL.

Solución: Los digramas descifrados con la matriz que habrá encontrado serán:

EC ⇒ WE;	TB ⇒ AL;	AZ ⇒ LX;	EN ⇒ LI;	WB ⇒ VE;
JH ⇒ IN;	TX ⇒ AY;	AB ⇒ EL;	BU ⇒ LO;	VC ⇒ WS;
LO ⇒ UB;	JT ⇒ MA;	PM ⇒ RI;	IL ⇒ NE.	

El texto en claro es M = WE ALL LIVE IN A YELLOW SUBMARINE.

■ ■ ■

2.4.4.2 Criptoanálisis del cifrado de Playfair

¿Qué puede decirse acerca de la distancia de unicidad del cifrador de *Playfair*? A simple vista, el poder ordenar aleatoriamente los 25 caracteres de la matriz, parece que la equivocación de la clave será el factorial de 25. No obstante, debido al algoritmo de cifra propuesto por *Playfair*, no todas las matrices de 5x5 son distintas. Lo veremos a continuación. Considérese la siguiente matriz de cifra con clave PATOS y la posterior rotación de las filas tres posiciones hacia abajo y, a continuación, una rotación de las columnas una posición hacia la izquierda:

P	A	T	O	S
B	C	D	E	F
G	H	I/J	K	L
M	N/Ñ	Q	R	U
V	W	X	Y	Z

Considere además las matrices recíprocas a esta:

G	H	I/J	K	L
M	N/Ñ	Q	R	U
V	W	X	Y	Z
P	A	T	O	S
B	C	D	E	F

H	I/J	K	L	G
N/Ñ	Q	R	U	M
W	X	Y	Z	V
A	T	O	S	P
C	D	E	F	B

Estas y otras matrices que se obtengan por rotaciones de filas y/o columnas son recíprocas de la primera puesto que se obtienen los mismos resultados al cifrar un texto según el método de *Playfair*. Por ejemplo, si se cifra con estas tres matrices el mensaje M = LA PATA Y EL PATO TUVIERON PATITOS, compruebe se obtiene el mismo criptograma C = HSAT OTOK GSTO SOMZ KDYE MATO QDSP. No obstante, esto sólo reduce el número de matrices posible en 4x4 = 16 que serán las recíprocas de la principal por lo que el valor de la entropía de la clave se asemeja mucho a la de un sistema del *César* con clave.

Por otra parte, el cifrado de *Playfair* presenta una debilidad que facilita el criptoanálisis. Conociendo qué palabras pueden ser comunes en el texto que se intenta romper, con la ayuda de las estadísticas de digramas comunes del lenguaje, se puede ir confeccionando la matriz y, finalmente, descifrar el criptograma. A continuación se comenta el procedimiento a seguir en este tipo de ataque; el lector interesado en profundizar sobre el tema puede consultar la referencia que se indica en la sección de la bibliografía recomendada para este capítulo.

En el lenguaje existen digramas más comunes que otros y como este sistema cifrará siempre el digrama M_1M_2 como el mismo criptograma C_1C_2, es lógico esperar una correspondencia de esta redundancia del lenguaje en el texto cifrado. Siguiendo una tabla de digramas, aparecen los siguientes pares con una frecuencia relativa superior a 500 para un texto con 40.000 caracteres.

Digrama	nº veces	Inverso	nº veces
DE	1084	ED	290
ES	1010	SE	547

Digrama	nº veces	Inverso	nº veces
EN	901	NE	370
OS	764	SO	212
AD	649	DA	436
TE	639	ET	115
IN	610	NI	191
ER	563	RE	537
AS	560	SA	227
EL	559	LE	245
OR	544	RO	372
NT	536	TN	24
ST	535	TS	19
RA	520	AR	493

Tabla 9. Digramas más frecuentes del lenguaje castellano y sus inversos.

Luego, siguiendo la matriz de cifrado con la clave PATOS se obtendrán, entre otros, los siguientes pares de cifrados:

M_1M_2	C_1C_2	M_1M_2	C_1C_2	M_1M_2	C_1C_2	M_1M_2	C_1C_2
DE	EF	ED	FE	ES	FO	SE	OF
EN	CR	NE	RC	ER	KY	RE	YK

Es decir, en este criptograma se verán representados todos los digramas del texto en claro, sólo que como digramas distintos; esto es, FO en vez de ES, YK en vez de RE, etc. Por lo tanto, si se observa que en el texto cifrado aparece, por ejemplo, el digrama XB con alta frecuencia y, simultáneamente, el digrama inverso BX con muy baja frecuencia, según la tabla 9 se podría suponer que se trata del digrama en claro NT o bien ST que cumplen con esta característica. Si se encuentra el digrama HK con alta frecuencia del texto cifrado y el inverso KH también tiene una frecuencia similar, podría tratarse de los digramas en claro ER o RA. El digrama más frecuente del criptograma podría ser DE, ES o EN en el texto en claro, etc.

Una vez determinadas algunas correspondencias, el criptoanalista también tiene en cuenta los caracteres que siguen a dichos digramas para formar trigramas y así ir estableciendo las letras equivalentes al alfabeto. Además, si conoce alguna palabra que supuestamente se repite en el texto en claro y que contenga entre sus caracteres las equivalencias anteriores, podrá encontrar más equivalencias y éstas otras más, como si se tratase de un procedimiento en cascada. Un análisis detallado del método está fuera del alcance de este libro; no obstante, siguiendo estas pautas y con mucha paciencia, este método a modo de un entretenido puzle nos lleva finalmente a la consecución de la matriz de cifra de *Playfair*.

2.4.4.3 Cifrador de Hill

En 1929, *Lester S. Hill*, un joven matemático, publica en Nueva York un artículo en el que propone la utilización del álgebra y, en particular de las matrices, en la operación de cifrado. La importancia del método de cifra propuesto por *Hill* descansa en la utilización de transformaciones lineales matriciales operando en módulo 26 -las letras del alfabeto inglés- con lo cual se facilita el cifrado

poligráfico, algo que tras el invento de *Playfair* fue insistentemente buscado por los criptólogos y matemáticos de la época. Desgraciadamente para *Hill*, su invento, aunque muy interesante para los científicos en aquella época, no era fácil de implantarlo en una máquina –no se había inventado el ordenador- y no pudo competir con otros criptógrafos que proliferaron en esos años como fueron la máquina *Enigma* de los alemanes y aparatos de cifra como los de *Hagelin*.

Dado que bajo ciertas condiciones este sistema presenta una alta seguridad, que puede implementarse fácilmente en los ordenadores actuales y que hace uso de una buena cantidad de conceptos de la aritmética modular operando con matrices, se profundizará en este cifrador y en su criptoanálisis. A pesar del alto valor de la posible entropía de su clave, verá que si se conoce el texto en claro y su criptograma asociado, este sistema no soporta un criptoanálisis.

Inicialmente, *Hill* plantea el problema como el conjunto de cuatro ecuaciones que se indican a continuación, en donde la variable y_i representa las letras cifradas y la variable x_i las letras del texto en claro.

$$
\begin{aligned}
y_1 &= 8\,x_1 + 6\,x_2 + 9\,x_3 + 5\,x_4 \bmod 26\\
y_2 &= 6\,x_1 + 9\,x_2 + 5\,x_3 + 10\,x_4 \bmod 26\\
y_3 &= 5\,x_1 + 8\,x_2 + 4\,x_3 + 9\,x_4 \bmod 26\\
y_4 &= 10\,x_1 + 6\,x_2 + 11\,x_3 + 4\,x_4 \bmod 26
\end{aligned}
$$

Por otra parte, *Hill* define un alfabeto de cifrado arbitrario, como el que se indica seguidamente, aunque aquí se usará el habitual: A = 0, B = 1, ... Z = 26.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
5	23	2	20	10	15	8	4	18	25	0	16	13	7	3	1	19	6	12	24	21	17	14	22	11	9

Figura 7. Alfabeto de cifrado propuesto por *Hill*.

El mensaje original utilizado por *Hill* era M = DELAY OPERATIONS. Toma entonces los cuatro primeros elementos, el tetragrama DELA, que corresponden a las variables x_1, x_2, x_3 y x_4, reemplaza sus valores de acuerdo al alfabeto indicado para obtener los primeros cuatro elementos del criptograma. Repite esta operación con los tetragramas restantes YOPE, RATI y ONS y con ello obtiene el criptograma C = JCOW ZLVB DVLE QMXC. Como en todo sistema poligrámico, se usan elementos de relleno si el último bloque tiene un tamaño menor. Para descifrar este criptograma, basta con resolver ahora el siguiente sistema de ecuaciones en x:

$$
\begin{aligned}
x_1 &= 23\,y_1 + 20\,y_2 + 5\,y_3 + 1\,y_4 \bmod 26\\
x_2 &= 2\,y_1 + 11\,y_2 + 18\,y_3 + 1\,y_4 \bmod 26\\
x_3 &= 2\,y_1 + 20\,y_2 + 6\,y_3 + 25\,y_4 \bmod 26\\
x_4 &= 25\,y_1 + 2\,y_2 + 22\,y_3 + 25\,y_4 \bmod 26
\end{aligned}
$$

La comprobación de la cifra completa del mensaje de *Hill* y su posterior recuperación se lo dejo como ejercicio para más adelante, pero por ahora se seguirá con la explicación de este método. De lo anterior, puede deducirse que el cifrado de *Hill* se trata de un cifrador por sustitución monoalfabética y poligrámico, en tanto que sustituye d caracteres del texto en claro por d caracteres de texto cifrado; en este caso d = 4. Si se representa el problema de *Hill* de cuatro ecuaciones mediante matrices, se tiene que la transformación para la operación de cifra será:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} = \begin{pmatrix} 8 & 6 & 9 & 5 \\ 6 & 9 & 5 & 10 \\ 5 & 8 & 4 & 9 \\ 10 & 6 & 11 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

Para la operación de descifrado se tendrá entonces:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 23 & 20 & 5 & 1 \\ 2 & 11 & 18 & 1 \\ 2 & 20 & 6 & 25 \\ 25 & 2 & 22 & 25 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}$$

Las operaciones anteriores pueden generalizarse suponiendo {y} = C, {x} = M y la matriz {K} de orden d*d como la clave K, luego:

$$C = K_E * M \bmod n$$
$$M = K_D * C \bmod n$$

En las ecuaciones anteriores, C y M serán vectores columna de dimensiones dx1, siendo *d* el tamaño del d-grama. En el caso del mensaje M, corresponderá al bloque de texto en claro de tamaño d-grama que se desea cifrar y para el criptograma C serán los d-grama elementos obtenidos al realizar la multiplicación de $\{K_E\}$ por {M} reduciendo los resultados módulo n. Observe que $\{K_D\}$ deberá ser la matriz inversa de la matriz de cifra $\{K_E\}$.

Como se ha comentado, al trabajar con bloques de información igual a d caracteres, es posible que el mensaje M no sea múltiplo de este valor. En tales circunstancias, se rellenará el último bloque hasta completar el d-grama con un carácter nulo que deberá ser conocido por quienes comparten el cifrado; por ejemplo la letra X. En cuanto al alfabeto de cifrado, si bien puede utilizarse cualquiera como lo propuso *Hill*, le recuerdo que se usa la representación numérica habitual en módulo 27, es decir, A = 0, B = 1, etc.

Consideraciones sobre la matriz K

La matriz K será siempre cuadrada, y sus elementos serán nuestra clave secreta. Será, además, el punto más importante del criptosistema, donde reside su seguridad. No será suficiente el hecho de que la clave sea una matriz cuadrada; ésta deberá cumplir ciertos requisitos:

a) Deberá ser una matriz de dimensión d*d, que viene dada por el d-grama que se desea cifrar. Puesto que en nuestro caso el d-grama a cifrar tiene d filas, la matriz clave deberá tener d columnas para que el producto sea factible. Por otra parte, como se desea que el resultado sea otro d-grama o matriz columna, entonces la matriz clave deberá tener d filas ya que el

resultado hereda el número de filas del primer factor de la multiplicación y el número de columnas del segundo factor.

b) Los elementos de la matriz serán enteros que formen parte del Conjunto Completo de Restos módulo n, en nuestro caso para el lenguaje castellano en mayúsculas [0, 26]. Utilizar números fuera de este rango no tiene sentido pues se estaría en una clase de equivalencia de dicho módulo. Una fórmula interesante de recordar los números que intervienen en la matriz clave, consiste en asignar letras a dichos números; en este caso dicha matriz con letras se trata de una matriz simbólica. A continuación se muestra un ejemplo de matriz simbólica para cifrado de trigramas con clave PELIGROSO.

$$K = \begin{pmatrix} P & E & L \\ I & G & R \\ O & S & O \end{pmatrix} = \begin{pmatrix} 16 & 4 & 11 \\ 8 & 6 & 18 \\ 15 & 19 & 15 \end{pmatrix}$$

La mayor utilidad de la matriz simbólica está en el intercambio de claves entre transmisor y receptor. Recuerde que en aquel entonces no había nacido aún la criptografía de clave pública; además, siempre es más humano recordar una clave como un conjunto de letras o palabras y no como un grupo de números sin sentido.

c) La matriz K no deberá ser singular, es decir, deberá tener inversa para poder realizar el proceso de descifrado. Para demostrar que una matriz no es singular, basta con demostrar que el determinante es distinto de cero.

Para encontrar la matriz inversa de K:

$$K^{-1} = \frac{T_{Adj(K)}}{|K|}$$

donde K^{-1} es la matriz Inversa de K, $T_{Adj(K)}$ es la Traspuesta de la matriz Adjunta de K, y $|K|$ es el determinante de K.

Aunque se supone un conocimiento básico de la aritmética matricial, a continuación se explican brevemente las operaciones necesarias para el cálculo de la matriz inversa. Dada una matriz K, su traspuesta $T_{(K)}$ será aquella en la que los elementos (i, j) se intercambian por los elementos (j, i), es decir se intercambian filas por columnas, como se indica en la siguiente ecuación:

$$\text{Si } K = \begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \text{ entonces } T_{(K)} = \begin{pmatrix} k_{11} & k_{21} & k_{31} \\ k_{12} & k_{22} & k_{32} \\ k_{13} & k_{23} & k_{33} \end{pmatrix}$$

Ejemplo: Encuentre la matriz trigrámica traspuesta de la matriz con clave simbólica K = NO ESTA MAL.

Solución: N = 13; O = 15; E = 4; S = 19; T = 20; A = 0; M = 12; A = 0; L = 11. Así:

$$K = \begin{pmatrix} 13 & 15 & 4 \\ 19 & 20 & 0 \\ 12 & 0 & 11 \end{pmatrix} \quad \text{luego} \quad T_{(K)} = \begin{pmatrix} 13 & 19 & 12 \\ 15 & 20 & 0 \\ 4 & 0 & 11 \end{pmatrix}$$

Si la matriz anterior sirve o no para cifrar mensajes en castellano módulo 27 (si está o no mal como dice el ejemplo) lo podrá comprobar un poco más adelante. Se llama matriz adjunta de K ($Adj_{(K)}$) a aquella que se obtiene al sustituir cada elemento a_{ij} por su adjunto correspondiente o, lo que es lo mismo, los determinantes de los elementos a_{ij} de la matriz K. Sea $|a_{ij}|$ dicho determinante, entonces:

$$K = \begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \quad Adj_{(K)} = \begin{pmatrix} |a_{11}| & -|a_{12}| & |a_{13}| \\ -|a_{21}| & |a_{22}| & -|a_{23}| \\ |a_{31}| & -|a_{32}| & |a_{33}| \end{pmatrix}$$

Para obtener $|a_{ij}|$ se elimina la fila i y la columna j y con los elementos que quedan se calcula su determinante. Por ejemplo, en la matriz el elemento $|a_{11}| = k_{22}*k_{33} - k_{32}*k_{23}$; $|a_{22}| = k_{11}*k_{33} - k_{31}*k_{13}$; etc.

Ejemplo: Para la matriz con los valores que se indican, se pide encontrar su matriz adjunta: $k_{11}=2$, $k_{12}=4$, $k_{13}=6$, $k_{21}=3$, $k_{22}=5$, $k_{23}=7$, $k_{31}=1$, $k_{32}=9$ y $k_{33}=0$.

Solución: La matriz en cuestión es:

$$K = \begin{pmatrix} 2 & 4 & 6 \\ 3 & 5 & 7 \\ 1 & 9 & 0 \end{pmatrix}$$

Luego los valores de los determinantes $|a_{ij}|$ son los siguientes:

$|a_{11}| = (5*0 - 9*7) = - 63$
$|a_{12}| = (3*0 - 1*7) = - 7$
$|a_{13}| = (3*9 - 1*5) = 22$
$|a_{21}| = (4*0 - 9*6) = - 54$
$|a_{22}| = (2*0 - 1*6) = - 6$
$|a_{23}| = (2*9 - 1*4) = 14$
$|a_{31}| = (4*7 - 5*6) = - 2$
$|a_{32}| = (2*7 - 3*6) = - 4$
$|a_{33}| = (2*5 - 3*4) = - 2$

Por lo tanto, la matriz adjunta de K, $Adj_{(K)}$ será:

$$Adj_{(K)} = \begin{pmatrix} |a_{11}| & -|a_{12}| & |a_{13}| \\ -|a_{21}| & |a_{22}| & -|a_{23}| \\ |a_{31}| & -|a_{32}| & |a_{33}| \end{pmatrix} = \begin{pmatrix} -63 & 7 & 22 \\ 54 & -6 & -14 \\ -2 & 4 & -2 \end{pmatrix}$$

Para ejercitarse un poco con estos cálculos, encuentre la matriz adjunta de la clave simbólica es K = ESTO NO ESTA TAN MAL.

Para poder encontrar la matriz inversa, deberá cumplirse que el determinante de ésta sea distinto de cero; en caso contrario la matriz es singular y, por tanto, no podrá ser utilizada como clave para cifrar. Ahora bien, puesto que se trabaja dentro de un cuerpo, esta condición de singularidad debe darse también dentro de él. Esto significa que el valor del determinante de la matriz clave reducido a módulo n tampoco debe ser igual a cero para que esta sea válida; es decir, deberá cumplirse que $|K| \bmod n \neq 0$ como se indica en el siguiente ejemplo.

Ejemplo: ¿Pueden utilizarse estas matrices para cifrar un mensaje en módulo 27?

$$K_1 = \begin{pmatrix} 1 & 2 & 2 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \qquad K_2 = \begin{pmatrix} 12 & 3 \\ 2 & 5 \end{pmatrix}$$

Solución: El determinante de K_1 para cifrar trigramas será igual a:

$|K_1| = k_{11}(k_{22}*k_{33} - k_{32}*k_{23}) - k_{12}(k_{21}*k_{33} - k_{31}*k_{23}) + k_{13}(k_{21}*k_{32} - k_{31}*k_{22})$

$|K_1| = 1(1*0 - 0*1) - 2(1*0 - 1*1) + 2(1*0 - 1*1) = 0$, luego $K_1 \bmod 27 = 0$.

La matriz de K_1 es singular y no puede ser usada para cifrar.

El determinante de K_2 para cifrar digramas será igual a:

$|K_2| = k_{11}*k_{22} - k_{21}*k_{12} = (12*5 - 2*3) = 54$

$|K_2| \bmod 27 = 54 \bmod 27 = 0$

La matriz de K2 tampoco puede ser usada para cifrar en módulo 27 al ser singular dentro del cuerpo.

Además del requisito $|K| \neq 0$, debe recordar que en criptografía se trabaja con números comprendidos entre 0 y n-1, el cuerpo de la cifra, por lo que no nos servirán los números fraccionarios. Para la existencia de la matriz inversa K^{-1} deberá ser posible 'dividir' por el determinante de K; es decir, debe existir el inverso de |K| en módulo n. Por lo tanto, dicha ecuación podrá escribirse como sigue:

$$K^{-1} = T_{Adj(K)} * inv(|K|, n) \bmod n$$

en donde se cumplirá que $inv(|K|,n) * |K| \bmod n = 1$. En realidad, este resultado nos permitirá encontrar la denominada matriz de identidad I que se verá más adelante. Como es bien sabido, el

valor de este inverso no siempre existe; la condición necesaria para su existencia es que |K| y el módulo n sean primos entre sí. Seguro que ya está convencido de esto.

Recapitulando entonces, en el cifrado de *Hill* deberá existir una transformación inversa que permita recuperar el mensaje en claro, y dicha operación sólo la puede dar la matriz clave. Esto quiere decir que las claves K_E y K_D deberán ser matrices inversas en el módulo de trabajo. Luego, la condición sine qua non del cifrador de *Hill* es que la matriz de cifrado tenga inversa, es decir:

$$K_D = K_E^{-1}$$

Ejemplo: a) Compruebe que la operación $[K] * [K^{-1}] = I$, en donde I es la matriz de identidad.

b) Compruebe este resultado en particular para la matriz K con elementos $k_{11} = 3$, $k_{12} = 2$, $k_{21} = 1$ y $k_{22} = 4$ en módulo 27.

Solución: a) Si $K = \begin{pmatrix} k_1 & k_2 \\ k_2 & k_2 \end{pmatrix}$ entonces $|K| = (k_1 \times k_2 - k_2 \times k_2)$

y la matriz inversa será:

$$K_{-1} = \frac{|K|}{I} \begin{pmatrix} -K^3 & K^1 \\ K^3 & -K^3 \end{pmatrix}$$

Si se multiplica $K^{-1} * K$ se tiene:

$$\frac{1}{|K|} \begin{pmatrix} k_2 & -k_2 \\ -k_2 & k_1 \end{pmatrix} \begin{pmatrix} k_1 & k_2 \\ k_2 & k_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I$$

b) Para la matriz indicada de 2x2 se tiene:

$$K = \begin{pmatrix} 3 & 2 \\ 1 & 4 \end{pmatrix} \bmod 27 \quad \text{Luego, } |K| = (3x4 - 1x2) \bmod 27 = 10$$

$$K^{-1} = \frac{1}{10} \begin{pmatrix} 4 & -2 \\ -1 & 3 \end{pmatrix} \bmod 27$$

Como inv(10,27) = 19 entonces:

$$K^{-1} = \begin{pmatrix} 4*19 & -2*19 \\ -1*19 & 3*19 \end{pmatrix} \bmod 27 = \begin{pmatrix} 22 & 16 \\ 8 & 3 \end{pmatrix} \bmod 27$$

Luego, multiplicando $K * K^{-1}$:

$$K*K^{-1} = \begin{pmatrix} 3 & 2 \\ 1 & 4 \end{pmatrix} \begin{pmatrix} 22 & 16 \\ 8 & 3 \end{pmatrix} \bmod 27 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \bmod 27$$

■■■

Los valores del producto de las matrices son:

$I_{11} = (3*22 + 2*8) \bmod 27 = (12 + 16) \bmod 27 = 1$

$I_{12} = (3*16 + 2*3) \bmod 27 = (21 + 6) \bmod 27 = 0$

$I_{21} = (1*22 + 4*8) \bmod 27 = (22 + 5) \bmod 27 = 0$

$I_{22} = (1*16 + 4*3) \bmod 27 = (16 + 12) \bmod 27 = 1$

Por último, al igual que en todos los sistemas que implican una multiplicación en el cifrado, para que exista la operación inversa debe cumplirse que el determinante de la matriz clave no sea múltiplo con el orden del grupo en el que se trabaja. Sólo si mcd[|K|,n] = 1 se podrá usar esa matriz K para cifrar. Luego, la condición suficiente y necesaria de esta matriz K será:

$$|K| \bmod n \neq 0$$
$$\text{mcd}[|K|, n] = 1$$

Ejemplo:¿Se puede utilizar esta matriz K para cifrar digramas en módulo 27 cuyos elementos son $k_{11} = 4$, $k_{12} = 3$, $k_{21} = 2$, $k_{22} = 6$?

Solución: Resolviendo el determinante de K:

$|K| = (k_{11}*k_{22}-k_{21}*k_{12}) = (4*6-2*3) = 18$. Como mcd(18,27) = 3, no puede usarse K como matriz de cifrado al carecer de inversa.

Con estos antecedentes, podrá entonces cifrar cualquier texto en claro mediante matrices usando una clave de dimensión d*d y agrupando el mensaje en bloques de tamaño d-gramas. Al igual que en otros sistemas, si es necesario se añaden al final del texto en claro caracteres de relleno para obtener el d-grama.

Ejemplo: Cifre el texto M = AMIGO CONDUCTOR, SI BEBES NO CONDUZCAS mediante trigramas usando la matriz simbólica con clave PELIGROSO.

Solución: Primero se comprobará que la matriz tiene inversa. El valor de |K| en módulo 27 es igual a 4 como se indica:

$|K| = [16(6*15 - 19*18) - 4(8*15 - 15*18) + 11(8*19 -15*6)] \bmod 27$

$|K| = [-9 + 6 + 7] \bmod 27 = 4$. Los valores de |K| y el módulo n son primos entre sí, luego la clave es válida.

El mensaje M se divide en trigramas genéricos del tipo $M_iM_jM_k$ como se indica: AMI GOC OND UCT ORS IBE BES NOC OND UZC ASX que pasan a cifrarse con la matriz de clave simbólica PELIGROSO. Puesto que (AMI) = (00 12 08) entonces el primer subcriptograma C_A será:

$$\begin{pmatrix} C_1 \\ C_2 \\ C_3 \end{pmatrix} = \begin{pmatrix} 16 & 4 & 11 \\ 8 & 6 & 18 \\ 15 & 19 & 15 \end{pmatrix} \begin{pmatrix} 0 \\ 12 \\ 8 \end{pmatrix} \bmod 27$$

Resolviendo esta matriz se obtiene:

$C_1 = (16*0 + 4*12 + 11*8) \bmod 27 = 136 \bmod 27 = 1$

$C_2 = (8*0 + 6*12 + 18*8) \bmod 27 = 216 \bmod 27 = 0$

$C_3 = (15*0 + 19*12 + 15*8) \bmod 27 = 348 \bmod 27 = 24$

$C_A = 01\ 00\ 27 = BAX$

Los demás subcriptogramas se los dejo como ejercicio.

Cifrador de Hill digrámico

Si un texto en claro $M = M_1M_2M_3M_4...M_N$ se cifra según el método de *Hill* en bloques de dos caracteres, para cada par de letras se tendrá:

$$C_1 = (k_{11}M_1 + k_{12}M_2) \bmod n$$

$$C_2 = (k_{21}M_1 + k_{22}M_2) \bmod n$$

$$\begin{pmatrix} C_1 \\ C_2 \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix} \begin{pmatrix} M_1 \\ M_2 \end{pmatrix} \bmod n$$

Ejemplo: Si la clave K es $k_{11} = 4$; $k_{12} = 2$; $k_{21} = 9$; $k_{22} = 2$, cifre el siguiente mensaje M = QUE TODA LA VIDA ES SUEÑO Y LOS SUEÑOS SUEÑOS SON.

Solución: $|K| = (k_{11}k_{22} - k_{21}k_{12}) \bmod 27 = (4*2 - 2*9) \bmod 27 = -10 \bmod 27$ que es igual a 17 mod 27. El mcd(17,27) = 1, luego existirá la matriz inversa.

Se representa el mensaje como se muestra y se procede a cifrar el primer digrama con la matriz K; los demás se los dejo como ejercicio.

M = QU ET OD AL AV ID AE SS UE ÑO YL OS SU EÑ OS SU EÑ OS SO NX

Como $[C_1C_2] = [K]x[M_1M_2]$, para $[M_1M_2]$ = QU = [17 21] se tiene:

$C_1 = (4*17+2*21) \bmod 27 = 2 = C$

$C_2 = (9*17+2*21) \bmod 27 = 6 = G$

C_1C_2 = CG. El criptograma completo que debe obtener es:

C = CGCV MGVV QQLX IIGT LIFU ÑEQL KXQK QLKX QKQL YMSD.

Para descifrar un criptograma de *Hill*, conocida la matriz clave, se calcula su inversa.

Ejemplo: Sea la matriz [K]: $k_{11}=2$, $k_{12}=10$, $k_{21}=17$, $k_{22}=5$. Se pide descifrar el siguiente criptograma C = NXXZ XSNX NEKE MJZT RVXD ÑZWB XZYW RJEV.

Solución: Cálculo de la matriz inversa K^{-1}:

$|K| = (2*5 - 17*10) \bmod 27 = 2$. El inv(2,27) = 14.

$$K^{-1} = \frac{1}{|K|}\begin{pmatrix} k_{22} & -k_{12} \\ -k_{21} & k_{11} \end{pmatrix} \bmod 27 = \frac{1}{2}\begin{pmatrix} 5 & -10 \\ -17 & 2 \end{pmatrix} \bmod 27$$

$$K^{-1} = \begin{pmatrix} 14*5 & -14*10 \\ -14*17 & 14*2 \end{pmatrix} \bmod 27$$

$$K^{-1} = \begin{pmatrix} 16 & 22 \\ 5 & 1 \end{pmatrix} \bmod 27$$

Aplicando la matriz inversa K^{-1} al criptograma C se obtiene el siguiente mensaje M = HILL SE HIZO FAMOSO PERO NO MILLONARIO.

Compruebe este resultado y si es capaz de hacerlo sin el auxilio de una calculadora, enhorabuena por su agilidad mental.

2.4.4.4 Criptoanálisis del cifrado de Hill

Si la matriz clave, en donde descansa la seguridad del sistema, no puede contener todas las combinaciones posibles del Conjunto Completo de Restos del módulo de trabajo, puesto que se eliminan aquellos que nos entregan un determinante igual a cero o bien tienen factores comunes con n, ¿qué tan seguro es este criptosistema? Se estudiarán a continuación el número de las matrices válidas para determinar la entropía de la clave H(K).

Suponga, por simplicidad, que se trabaja en módulo 2, posteriormente se hará en módulo 27. Si la matriz es de orden dos, es decir, tiene los elementos k_{11}, k_{12}, k_{21} y k_{22}, entonces el número posible de matrices será igual a 16 puesto que existirán 2^4 combinaciones del CCR módulo 2, es decir los valores 0 y 1. Estas matrices serán:

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}; \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}; \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}; \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}; \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}; \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}; \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}; \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}; \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}; \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}; \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}; \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}; \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}; \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}; \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

No obstante, solamente existirán 6 matrices válidas, aquellas en las que el determinante es distinto de cero; en este caso en particular las matrices 7ª y 8ª de la primera fila y 2ª, 4ª, 6ª y 7ª de la segunda. Si, por ejemplo, se utilizan sólo tres números, los restos 0, 1 y 2 del módulo 27 como elementos de la matriz, se obtienen $3^4 = 81$ matrices de 2x2. De estas matrices 33 serán no válidas por lo que sólo 48 matrices clave con los restos 0, 1 y 2 permiten cifrar digramas en dicho cuerpo. Si se atreve y tiene tiempo suficiente, compruébelo.

En el valor de módulo 27, habrá $27^4 = 531.441$ matrices distintas de orden 2 cuyos elementos son el CCR(27), es decir [0, 26]. Si ahora se descartan aquellas matrices en las que el determinante es igual a cero o bien tienen factor común con el módulo 27, el número de matrices válidas se reduce a 314.928. Si se quiere aumentar la entropía de la clave, podría trabajar con un módulo primo, por

ejemplo en módulo 37 con un alfabeto de letras mayúsculas más los dígitos del 0 al 9, de forma que en este caso prácticamente sólo se eliminen aquellas matrices cuyo determinante sea cero. Con un módulo igual a 37, el número de matrices 2x2 crece hasta 1.874.161 de las que más de 1.800.000 son claves válidas.

En cuanto a la distancia de unicidad de este cifrador, que dependerá de la entropía de la clave o, lo que es lo mismo, del número de matrices válidas para cifrar, puede aproximarse de forma empírica para digramas y trigramas los siguientes valores cuando el módulo de trabajo n es un número primo:

Digramas: $N = H(K)/D \approx \log_2[n^4 - n^3 - n^2 + n]/3,4$

Trigramas: $N = H(K)/D \approx \log_2[n^9 - n^8 - n^7 + n^5 + n^4 - n^3]/3,4$

Por ejemplo, cifrando digramas en módulo 37 se obtendría una distancia de unicidad igual a 6,1 caracteres; aumentando el d-grama en una unidad, es decir cifrando trigramas, la distancia de unicidad crece hasta 13,8 que es más del doble de la anterior. Como el número de matrices tiende a $n(\exp d^2)$ para poligramas de longitud d, la distancia de unicidad aumentará significativamente. Por desgracia, esta característica no aumentará su nivel de seguridad como se verá más adelante.

El cifrador de *Hill* se muestra por tanto, al menos en una primera aproximación, bastante robusto ante un ataque puesto que, además, el algoritmo de cifrado destruye las estadísticas del lenguaje y cuenta con una característica muy interesante en criptografía: el cifrado de los caracteres de un bloque dependerá también de los caracteres que forman el poligrama y de su posición relativa en él. Por ejemplo, si se cifran dos mensajes M_A = OK y M_B = OH, en principio muy similares aunque signifiquen cosas distintas, el resultado no tendrá en cuenta para nada esta relación. Por ejemplo, si la matriz de cifra es $k_{11} = 2$, $k_{12} = 1$, $k_{21} = 1$ y $k_{22} = 3$, estos mensajes se cifrarán como sigue:

M_A = OK ⇒ $C_1 = (15*2 + 10*1) \bmod 27 = 13$ = N
$C_2 = (15*1 + 10*3) \bmod 27 = 18$ = R
Luego C = NR

M_B = OH ⇒ $C_1 = (15*2 + 7*1) \bmod 27 = 10$ = K
$C_2 = (15*1 + 7*3) \bmod 27 = 9$ = J
Luego C = KJ

Por otra parte, si M = KO el criptograma será C = IB. Luego, un simple cambio de posición de los caracteres en el texto en claro o modificaciones mínimas, producen una alteración total en el criptograma. Esto se debe a la ecuación genérica de cifra que, por ejemplo, para digramas es:

$$C_i = (k_{i1}*M_1 + k_{i2}*M_2) \bmod n$$

De lo anterior se deduce que el carácter que ocupa la posición iésima en el criptograma depende no sólo del carácter que ocupa la posición iésima en el texto en claro, sino también del siguiente en la posición iésima+1 que conforma, en este caso, el digrama. Luego, mientras mayor sea el tamaño del poligrama utilizado, cada carácter dependerá de más caracteres del texto en claro y ahí radica en principio la fortaleza de este cifrado. En estas condiciones, no cabe plantearse un ataque por análisis de frecuencias. Por otra parte, el ataque por fuerza bruta puede ser extremadamente difícil si se elige un primo como módulo de trabajo y se cifran bloques de texto de un tamaño igual o mayor que 5.

■ ■ ■

Ejemplo: Si el grupo de trabajo es el primo 37 y por tanto casi el 100% de las matrices de cifrado son válidas, ¿qué tamaño de poligrama debe usarse para que la entropía de la clave H(K) del cifrado de *Hill* sea del orden de la del algoritmo DES igual a 56?

Solución: Para d=3, la matriz clave K tendrá 3x3=9 elementos, por lo que el número de matrices puede estimarse en $37^9 \approx 1{,}3 \times 10^{14}$. Luego la entropía de la clave $H(K) = \log_2(1{,}3 \times 10^{14}) = 46{,}9$. Aumentando a tetragramas, el número de matrices es del orden de $37^{16} \approx 1{,}2 \times 10^{25}$ con lo que la entropía en este caso se eleva a $H(K) = \log_2(1{,}2 \times 10^{25}) = 83{,}3$. Luego, para un poligrama igual a 4 caracteres, este cifrado tendría una fortaleza similar al DES estándar en cuanto a la distancia de unicidad. ¿Y si ahora se cifran con bloques de texto en claro de 8 caracteres (64 bits) como lo hace el DES?

Como es fácil apreciar, en estas condiciones un ataque por fuerza bruta es impensable. La única posibilidad de ataque a este tipo de cifra es la elección de un texto en claro y buscar vectores unitarios en el mensaje o en el criptograma, y en el caso de no encontrarlos aplicar el método de *Gauss-Jordan* contando ahora sólo con un criptograma y su correspondiente texto en claro. En cualquier caso se supondrá, además, que el criptoanalista conoce que el cifrado se trata de *Hill*, que conoce el tamaño del poligrama usado para la cifra y la correspondencia entre los caracteres del alfabeto en claro y su equivalente numérico. He aquí el verdadero Talón de Aquiles de este cifrador y la razón por la que, incluso alcanzando un valor de distancia de unicidad muy alto, no es seguro y por tanto cayó en desuso.

Ataque con elección del texto en claro o criptograma

¿Qué es eso de los vectores unitarios y el ataque aplicando el método de *Gauss-Jordan*? Se explicará, a continuación. Suponga que el criptoanalista cuenta con el criptograma y los correspondientes textos en claro de varios mensajes. Luego, podrá elegir bloques específicos que le reporten mayor información. Para este tipo de cifra interesa encontrar los vectores unitarios de la dimensión en la que se trabaja. El resultado de cifrar estos poligramas serán los distintos valores de las columnas de la matriz clave como comprobará ahora mismo.

Un vector unitario de dimensión n es aquel que tiene todos sus elementos nulos excepto el elemento i-ésimo que es la unidad. Por ejemplo, para una cifra con trigramas, n = 3, la matriz de Identidad I_3 tendrá los vectores unitarios μ_1, μ_2 y μ_3 que se indican:

$$\text{Si } I_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{entonces} \quad \begin{matrix} \mu_1 = [1\,0\,0] \\ \mu_2 = [0\,1\,0] \\ \mu_3 = [0\,0\,1] \end{matrix}$$

Suponga entonces que realiza la siguiente operación matricial:

$$K \times \mu_1 = \begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} k_{11} \\ k_{21} \\ k_{31} \end{pmatrix}$$

Como se observa, resolviendo la ecuación se obtiene la primera columna de la matriz clave. Si se realiza la misma operación con los vectores unitarios μ_2 y μ_3 encontrará la segunda y tercera columnas de dicha matriz. Por lo tanto, al cifrar con un vector unitario i, se encuentra la columna i de la matriz clave que precisamente será el trigrama de texto cifrado que conoce el criptoanalista pues K ∗ M = C. Si la codificación del alfabeto de cifrado es el habitual módulo 27 entonces los vectores unitarios para este caso trigrámico serán:

$$\mu_1 = [\,B\ A\ A\,]$$
$$\mu_2 = [\,A\ B\ A\,]$$
$$\mu_3 = [\,A\ A\ B\,]$$

Ejemplo: Si se ha recibido el criptograma C y se conoce que pertenece al mensaje en claro M, se pide encontrar la matriz clave de cifrado digrámico.

M = EL BANDIDO FUE ABATIDO AL ATARDECER.

C = OYFCQ LSBEW FECEN ZSBUD BVSNO UXPCZ.

Solución: Se escribe el mensaje y el criptograma en digramas:

M = EL BA ND ID OF UE AB AT ID OA LA TA RD EC ER.

C = OY FC QL SB EW FE CE NZ SB UD BV SN OU XP CZ.

Aparecen los vectores unitarios [A B] y [B A] en los digramas segundo y séptimo. El cifrado correspondiente al vector [B A] es FC, es decir los números 5 y 2, en tanto que el correspondiente al vector [A B] es CE, es decir 2 y 4. Luego la matriz clave será:

$$K = \begin{pmatrix} 5 & 2 \\ 2 & 4 \end{pmatrix}$$

En el ejemplo anterior, para el primer digrama de texto en claro BA = [1 0] se tiene que $C_1 = k_{11}*1 + k_{12}*0 = k_{11} = 5$ = F y $C_2 = k_{21}*1 + k_{22}*0 = k_{21} = 2$ = C, el digrama que aparece en el texto cifrado. Por lo tanto, si en el texto en claro el criptoanalista encuentra estas cadenas de vectores unitarios, será capaz de encontrar la matriz de cifrado. Esto será también válido para cifrados trigrámicos. No obstante, para n mayor que tres, el método deja de ser válido pues existen pocas cadenas de ese tipo en castellano. El vector de longitud cuatro AAAB podría encontrarse en el texto "... así que ella estaba dispuesta a abanicarse por el calor que hacía..." pero esto es hilar demasiado fino porque nos faltaría encontrar otros vectores como BAAA, ABAA, y AABA. Para cinco letras, seguro que no existen.

Si sólo se encuentra un vector unitario, por ejemplo [A B], sigue siendo posible descifrar la matriz clave digrámica. De igual manera sucederá si para un cifrado trigrámico aparecen dos vectores unitarios, por ejemplo [A A B] y [A B A]. El restante vector se puede deducir aplicando la ecuación C = K ∗ M con las incógnitas del caso; como se conocen C y M, se pueden despejar las incógnitas de la columna de la matriz clave que falta.

Ejemplo: El mensaje M = "Ese abanico estaba abajo" se cifra por trigramas según *Hill* y se obtiene el siguiente criptograma:

M = ESE ABA NIC OES TAB AAB AJO

C = AEA DFI EJL KTL QYÑ EGJ GAR

Se pide encontrar la matriz clave.

Solución: En el texto en claro existen dos vectores unitarios, el vector μ_2 = [ABA] que se cifra como [DFI] por lo que la segunda columna de la matriz clave será [3 5 8] y luego μ_3 = [AAB] que se cifra como [EGJ] y por tanto la tercera columna de la matriz será [4 6 9]. Luego se tendrá la siguiente matriz:

$$K = \begin{pmatrix} k_{11} & 3 & 4 \\ k_{21} & 5 & 6 \\ k_{31} & 8 & 9 \end{pmatrix}$$

Tomando por ejemplo el primer trigrama, se tiene que el texto en claro [ESE] = [4 19 4] se cifra como [AEA] = [0 4 0], luego se cumplirá que:

$$\begin{pmatrix} 0 \\ 4 \\ 0 \end{pmatrix} = \begin{pmatrix} k_{11} & 3 & 4 \\ k_{21} & 5 & 6 \\ k_{31} & 8 & 9 \end{pmatrix} x \begin{pmatrix} 4 \\ 19 \\ 4 \end{pmatrix}$$

Resolviendo:

$$0 = (k_{11}*4 + 3*19 + 4*4) \bmod 27 \Rightarrow k_{11} = 2$$

$$4 = (k_{21}*4 + 5*19 + 6*4) \bmod 27 \Rightarrow k_{21} = 5$$

$$0 = (k_{31}*4 + 8*19 + 9*4) \bmod 27 \Rightarrow k_{31} = 7$$

(Véase la explicación a continuación del ejemplo)

Luego la matriz clave K será:

$$K = \begin{pmatrix} 2 & 3 & 4 \\ 5 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

La elección del texto en claro [ESE] = [4 19 4] en el ejemplo anterior es la adecuada puesto que los valores de k_{11}, k_{21} y k_{31} se obtendrán al multiplicarse por el primer elemento, la letra E = 4, que tiene inverso en n. Puesto que inv(4,27) = 7:

k_{11}: $0 = (k_{11}*4 + 73) \bmod 27 = (k_{11}*4 + 19) \bmod 27$
$k_{11} = (0 - 19) *\text{inv}(4,27) \bmod 27 = 8*7 \bmod 27 = 56 \bmod 27 = 2 \Rightarrow k_{11} = 2$

k_{21}: $4 = (k_{21}*4 + 119) \bmod 27 = (k_{21}*4 + 11) \bmod 27$
$k_{21} = (4 - 11) *\text{inv}(4,27) \bmod 27 = 20*7 \bmod 27 = 140 \bmod 27 = 5 \Rightarrow k_{21} = 5$

k_{31}: $0 = (k_{31}*4 + 188) \bmod 27 = (k_{31}*4 + 26) \bmod 27$
$k_{31} = (0 - 26) *\text{inv}(4,27) \bmod 27 = 1*7 \bmod 27 = 7 \bmod 27 = 7 \Rightarrow k_{31} = 7$

A igual resultado se llega al tomar en este ejemplo los pares mensaje criptograma NIC/EJL y TAB/QYÑ; no así si la elección es OES/KTL y AJO/GAR.

Ejemplo: Demuestre que se obtiene la misma primera columna de la matriz k_{11}, k_{21} y k_{31} del ejemplo anterior, eligiendo el par mensaje/criptograma NIC/EJL y que la elección de los pares EOS/KTL y AJO/GAR no es la adecuada.

Solución: a) El mensaje NIC tiene el equivalente numérico 13, 8, 2 y el criptograma EJL 4, 9, 11. Luego:

$$\begin{pmatrix} 4 \\ 9 \\ 11 \end{pmatrix} = \begin{pmatrix} k_{11} & 3 & 4 \\ k_{21} & 5 & 6 \\ k_{31} & 8 & 9 \end{pmatrix} x \begin{pmatrix} 13 \\ 8 \\ 2 \end{pmatrix}$$

$4 = (k_{11}*13 + 3*8 + 4*2) \bmod 27$

$k_{11} = (4 - 32) *inv(13,27) \bmod 27 = 26*25 \bmod 27 = 650 \bmod 27 = 2$

$9 = (k_{21}*13 + 5*8 + 6*2) \bmod 27$

$k_{21} = (9 - 52) *inv(13,27) \bmod 27 = 11*25 \bmod 27 = 275 \bmod 27 = 5$

$11 = (k_{31}*13 + 8*8 + 9*2) \bmod 27$

$k_{31} = (11 - 82) *inv(13,27) \bmod 27 = 10*25 \bmod 27 = 250 \bmod 27 = 7$

b) Para el par EOS/KTL se tiene:

$$\begin{pmatrix} 10 \\ 20 \\ 11 \end{pmatrix} = \begin{pmatrix} k_{11} & 3 & 4 \\ k_{21} & 5 & 6 \\ k_{31} & 8 & 9 \end{pmatrix} x \begin{pmatrix} 15 \\ 4 \\ 19 \end{pmatrix}$$

$10 = (k_{11}*15 + 3*4 + 4*19) \bmod 27$

$k_{11} = (10 - 88) *inv(15,27) \bmod 27$

Como mcd(15,27) = 3, no existe inverso y no puede calcularse k_{i1}.

c) Para el par AJO/GAR se tiene:

$$\begin{pmatrix} 6 \\ 0 \\ 18 \end{pmatrix} = \begin{pmatrix} k_{11} & 3 & 4 \\ k_{21} & 5 & 6 \\ k_{31} & 8 & 9 \end{pmatrix} x \begin{pmatrix} 0 \\ 9 \\ 15 \end{pmatrix}$$

$6 = (k_{11}*0 + 3*9 + 4*15) \bmod 27$

$k_{11} = (6 - 87) *inv(0,27) \bmod 27$

Como no existe inv(0,n) no puede calcularse k_{i1}.

En el ejemplo anterior, se podría pensar abordar el punto b) mediante el método de prueba de valores de k_{11} en la ecuación $10 = (k_{11}*15 + 88) \bmod 27$. No obstante, esto es un error como se verá a

continuación. Evidentemente el valor k_{11}=2 (que es el valor verdadero) cumple con la ecuación anterior pero también se cumple dicha ecuación para los valores k_{11} = 11 y k_{11} = 20 lo cual no tiene sentido porque la solución debe ser única.

Ahora bien, si el texto en claro no cuenta con estos vectores unitarios, también puede buscarlos en el criptograma. En este caso, el procedimiento nos lleva a recuperar la matriz inversa de la utilizada para cifrar.

Ejemplo: Se tiene el siguiente texto en claro y su criptograma que se sabe ha sido cifrado mediante *Hill* por digramas. Encuentre la matriz clave.

M = HILL SE PUEDE ATACAR Y ROMPER LA CIFRA BUSCANDO VECTORES

C = IBSD WJ QQLCL QBESVA B JOXHLI KN BAHSR EOKSVCTM KTYBKAYFI

Solución: Ordenando por digramas:

M = HI LL SE PU ED EA TA CA RY RO MP ER LA CI FR AB US CA ND OV EC TO RE SX

C = IB SD WJ QQ LC LQ BE SV <u>AB</u> JO XH LI KN <u>BA</u> HS RE OK SV CT MK TY BK AY FI

En el criptograma están los dos vectores unitarios [BA] = [1 0] con su par en el texto en claro [CI] = [2 8] y el vector [AB] = [0 1] con su par de texto en claro [RY] = [18 25]. Luego, se tiene que:

$$K^{-1} = \begin{pmatrix} 2 & 18 \\ 8 & 25 \end{pmatrix}$$

Como $K = (K^{-1})^{-1}$, se puede deducir que la matriz clave será entonces:

$$K = \begin{pmatrix} 23 & 18 \\ 11 & 4 \end{pmatrix}$$

Con esta matriz de cifra K se obtiene el criptograma indicado.

Para el caso de cifra con digramas, se busca algún digrama que contenga el valor cero o la letra A, bien en el mensaje en claro o bien en el criptograma; planteándose entonces un sistema de ecuaciones en donde la única condición a cumplir es que el elemento que acompañe a esa letra A tenga inverso en el cuerpo de cifra. De esta manera se obtiene una de las columnas de la matriz clave. Para encontrar la columna restante se busca otra ecuación de cifra, en donde el elemento que multiplica a los k_{ij} deberá también tener inverso, como se mostrará en el siguiente ejemplo. ¿Y si no se cuenta con estos vectores unitarios? Aunque no lo crea, todavía puede atacarse al sistema si el texto en claro es conocido.

Ejemplo:Se nos pide realizar un ataque al sistema de cifra de *Hill* digrámico según el método explicado. El texto en claro y su criptograma son:

M = HABIA VIDA EN MARTE

C = PIEBX PQYX YN FARIQ

Solución: Agrupando texto en claro y criptograma por digramas:

M = HA BI AV ID AE NM AR TE

C = PI EB XP QY XY NF AR IQ

Como no se aprecian vectores unitarios por ninguna parte, se planteará la primera ecuación de cifra para el primer digrama HA en donde aparece la letra H = 7 en el texto en claro acompañado de la letra A = 0. Puesto que inv (7,27) = 4, entonces:

$$\begin{pmatrix} P \\ I \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix} x \begin{pmatrix} H \\ A \end{pmatrix} \qquad \begin{pmatrix} 16 \\ 8 \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix} x \begin{pmatrix} 7 \\ 0 \end{pmatrix}$$

$16 = k_{11} * 7 \Rightarrow k_{11} = 16 * inv(7,27) \bmod 27 = 16 * 4 \bmod 27 = 10$
$8 = k_{21} * 7 \Rightarrow k_{21} = 8 * inv(7,27) \bmod 27 = 8 * 4 \bmod 27 = 5$

La ecuación del digrama (XP) = K ∗ (AV), en donde V = 22 tiene como inverso 16, nos entrega la siguiente matriz:

$$\begin{pmatrix} X \\ P \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix} x \begin{pmatrix} A \\ V \end{pmatrix} \qquad \begin{pmatrix} 24 \\ 16 \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix} x \begin{pmatrix} 0 \\ 22 \end{pmatrix}$$

$24 = k_{12} * 22 \Rightarrow k_{12} = 24 * inv(22,27) \bmod 27 = 24 * 16 \bmod 27 = 6$
$16 = k_{22} * 22 \Rightarrow k_{22} = 16 * inv(22,27) \bmod 27 = 16 * 16 \bmod 27 = 13$

Por lo tanto, la matriz de cifra del ejemplo es:

$$K = \begin{pmatrix} 10 & 6 \\ 5 & 13 \end{pmatrix}$$

Este método podría generalizarse para matrices de mayor rango, aunque como es lógico aumentará la dificultad de encontrar poligramas con todos los elementos excepto uno iguales a cero, por lo que este método resulta poco práctico. La generalización del ataque anterior mediante el planteamiento de un sistema de ecuaciones matriciales se conoce como método de *Gauss-Jordan* y será tratado en el próximo apartado. En este otro escenario prácticamente no hay criptograma que se resista a este ataque; no obstante, si se desconoce el texto en claro, debido al alto valor de la distancia de unicidad de este cifrador resulta absurdo intentar un ataque por fuerza bruta.

Ataque con texto en claro conocido

Al no poder utilizar la técnica anterior porque no se encuentran los vectores unitarios en el texto en claro o en el criptograma, el criptoanalista siempre podrá atacar un cifrado de *Hill* si cuenta, por lo menos, con un criptograma y su texto en claro asociado. Siguiendo el método propuesto por *Alan Konheim* en “Cryptography: A Primer” el procedimiento consiste en disponer las correspondencias entre el texto en claro y el texto cifrado en forma de matriz y utilizar el método de *Gauss-Jordan* que consiste, básicamente, en aplicar operaciones elementales a la matriz hasta conseguir (si se puede) “diagonalizar” la parte izquierda, de forma que la diagonal principal sea la unidad. Esto quiere decir

que en la mitad izquierda estarán los vectores unitarios que fueron definidos en el apartado anterior, por lo que la otra mitad derecha tendrá una relación directa con la matriz clave buscada.

Si la matriz 2n-grámica se define como [(TextoEnClaro) | (TextoCifrado)] la parte derecha, una vez diagonalizada la izquierda, será la traspuesta de la matriz clave de cifrado K. Le dejo aquí un nuevo ejercicio: definir ahora la matriz 2n-grámica como [(TextoCifrado) | (TextoEnClaro)] y una vez diagonalizada la parte izquierda comprobar qué tipo de matriz se obtiene. Por ejemplo, suponga que tiene el siguiente texto en claro asociado con el criptograma de *Hill* trigrámico que se indica:

```
M = ENU NLU GAR DEL AMA NCH ADE CUY ONO MBR ...
C = WVX IDQ DDO ITQ JGO GJI YMG FVC UÑT RLL ...
```

Que representado a modo de ejemplo la una matriz 2n-grámica de *Gauss-Jordan* se obtiene.

$$\begin{matrix} \textit{MatrizTrigrámica} \\ \textit{Texto en Claro} \\ \textit{Texto Cifrado} \end{matrix} \begin{pmatrix} E & N & U & W & V & X \\ N & L & U & I & D & Q \\ G & A & R & D & D & O \\ D & E & L & I & T & Q \\ A & M & A & J & G & O \\ N & C & H & G & J & I \\ A & D & E & Y & M & G \\ C & U & Y & F & V & C \\ O & N & O & U & Ñ & T \\ M & B & R & R & L & L \end{pmatrix} = \begin{pmatrix} 4 & 13 & 21 & 23 & 22 & 24 \\ 13 & 11 & 21 & 8 & 3 & 17 \\ 6 & 0 & 18 & 3 & 3 & 15 \\ 3 & 4 & 11 & 8 & 20 & 17 \\ 0 & 12 & 0 & 9 & 6 & 15 \\ 13 & 2 & 7 & 6 & 9 & 8 \\ 0 & 3 & 4 & 25 & 12 & 6 \\ 2 & 21 & 25 & 5 & 22 & 2 \\ 15 & 13 & 15 & 21 & 14 & 20 \\ 12 & 1 & 18 & 18 & 11 & 11 \end{pmatrix}$$

Como se observa, no aparecen vectores trigrámicos unitarios ni en el texto en claro ni en el texto cifrado por lo que se intentará el ataque por *Gauss-Jordan*. Se escriben entonces los trigramas del texto en claro a la izquierda y los del criptograma a la derecha en una matriz 2 n-grámica, con los correspondientes equivalentes numéricos.

El primer paso será conseguir que toda la primera columna sea 0 excepto el elemento a_{11}. Para ello se multiplica la fila primera por 7 ya que inv (4, 27) = 7 con lo que se tiene (4*7 13*7 21*7 23*7 22*7 24*7) mod 27 = (1 10 12 26 19 6).

Si el primer elemento de la fila (en este caso E = 4) tuviera algún factor común con el módulo 27, el método sigue siendo válido porque en ese caso se moverían filas enteras y alguna habrá cuyo primer elemento sea primo relativo con el módulo. La matriz no cambiará; es más, a nivel matemático da lo mismo donde estén localizadas las filas. Hecho esto se realizan las siguientes operaciones básicas módulo 27:

a) 2ª fila = 2ª fila - 13 * 1ª fila

b) 3ª fila = 3ª fila - 6 * 1ª fila

c) 4ª fila = 4ª fila - 3 * 1ª fila

d) 6º fila = 6ª fila - 13 * 1ª fila

e) 8ª fila = 8ª fila - 2 ∗ 1ª fila
f) 9ª fila = 9ª fila - 15 ∗ 1ª fila
g) 10ª fila = 10ª fila - 12 ∗ 1ª fila

Se obtiene entonces la siguiente matriz:

$$\begin{matrix} \\ \\ \\ \\ \textit{MatrizTrigrámica} \\ \textit{Texto en Claro} \\ \textit{Texto Cifrado} \\ \\ \\ \\ \end{matrix} \quad \begin{pmatrix} 1 & 10 & 12 & 26 & 19 & 6 \\ 0 & 16 & 0 & 21 & 26 & 20 \\ 0 & 21 & 0 & 9 & 24 & 6 \\ 0 & 1 & 2 & 11 & 17 & 26 \\ 0 & 12 & 0 & 9 & 6 & 15 \\ 0 & 7 & 13 & 19 & 5 & 11 \\ 0 & 3 & 4 & 25 & 12 & 6 \\ 0 & 1 & 1 & 7 & 11 & 17 \\ 0 & 25 & 24 & 9 & 26 & 11 \\ 0 & 16 & 9 & 3 & 26 & 20 \end{pmatrix}$$

Se procede de igual manera con las columnas segunda y tercera. Observe además que en el cálculo de la columna tercera se ha tenido que mover filas porque aparece el valor 0 en el tercer elemento. Como ejercicio, compruebe qué movimientos se han hecho.

Al final de todo el proceso se obtiene la matriz 2n-grámica que se indica en donde se observan los vectores unitarios en la matriz de la izquierda, correspondiente al texto en claro.

$$\begin{matrix} \\ \\ \textit{MatrizTrigrámica} \\ \textit{Texto en Claro} \\ \textit{Texto Cifrado} \\ \textit{diagonalizada} \\ \textit{agrupando los} \\ \textit{vectores unitarios} \\ \\ \end{matrix} \quad \begin{pmatrix} 1 & 0 & 0 & 2 & 5 & 7 \\ 0 & 1 & 0 & 3 & 5 & 8 \\ 0 & 0 & 1 & 4 & 6 & 9 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Como la mitad izquierda correspondía al texto en claro, la parte derecha de la matriz con vectores unitarios será la traspuesta de la clave. Esto es, 100 es el primer vector unitario y entonces entrega la primera columna de la matriz de clave (2, 5, 7); y de igual manera sucede con los vectores segundo 010 (3, 5, 8) y tercero 001 (4, 6, 9). Luego, la clave será:

$$K = \begin{pmatrix} 2 & 3 & 4 \\ 5 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Pasará a comprobarse que ésta es la matriz verdadera cifrando el primer trigrama del mensaje M = [ENU] = [4 13 21] que debe darnos el trigrama C = [WVX] = [23 22 24].

$$C = K x M = \begin{pmatrix} 2 & 3 & 4 \\ 5 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} x \begin{pmatrix} 4 \\ 13 \\ 21 \end{pmatrix}$$

$$C_1 = [2*4 + 3*13 + 4*21] \bmod 27 = 23 = W$$
$$C_2 = [5*4 + 5*13 + 6*21] \bmod 27 = 22 = V$$
$$C_3 = [7*4 + 8*13 + 9*21] \bmod 27 = 24 = X$$

Le dejo como ejercicio comprobar la cifra completa del mensaje de este ejemplo y deducir la clave a partir de [(TextoCifrado) | (TextoEnClaro)].

2.5 Cifradores por transposición

El segundo método clásico utilizado para cifrar mensajes es la transposición o permutación de caracteres. Esto consiste en reordenar los caracteres del texto en claro como si de una baraja de cartas se tratase. El resultado de tal acción es la de difuminar la información del texto en claro y provocar, por tanto, la difusión propuesta por *Shannon* para la protección de la misma. Precisamente este método era el utilizado por los lacedemonios en el sistema de cifra de la escítala, un cifrador de la categoría de transposición por grupos.

En el siglo V antes de J.C. los lacedemonios, un antiguo pueblo griego, usaban el método de la escítala para cifrar sus mensajes. El sistema consistía en una cinta que se enrollaba en un bastón y sobre el cual se escribía el mensaje en forma longitudinal, como se aprecia en la figura.

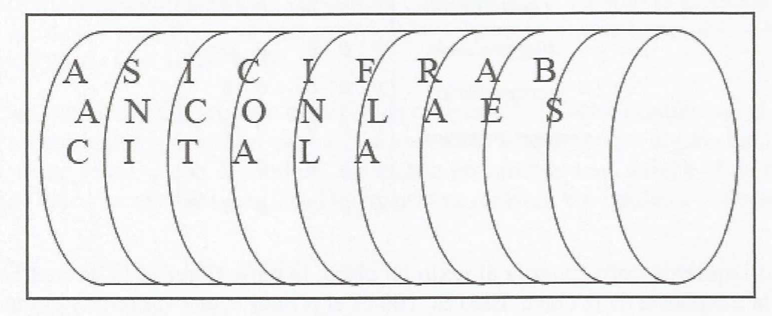


Figura 8. Mensaje ocultado con la escítala.

Una vez escrito el mensaje, la cinta se desenrollaba y era entregada al mensajero. Si éste era interceptado por cualquier enemigo, lo único que se conseguía era una cinta de cuero con un conjunto

de caracteres o letras distribuidas al parecer de forma aleatoria. Incluso si el enemigo intentara enrollar la cinta en un bastón con diámetro diferente, el resultado obtenido sería un conjunto de letras escritas una a continuación de otra sin sentido alguno.

Por ejemplo, en el caso de la figura anterior, la cinta llevará el mensaje M = ASI CIFRABAN CON LA ESCITALA si bien en ella sólo podrá leerse el criptograma C = AACSNIICTCOAINLFLARAAEBS. Para enmascarar completamente la escritura, es obvio que la cinta en cuestión debe tener caracteres en todo su contorno. Como es de esperar, la clave del sistema residía precisamente en el diámetro de aquel bastón, de forma que solamente el receptor autorizado tenía una copia exacta del mismo bastón en el que enrollaba el mensaje recibido y, por tanto, podía leer el texto en claro.

En este sistema no existe modificación alguna del mensaje; es decir, los caracteres van en claro desde el transmisor hacia el receptor, por lo que como se verá más adelante se tratará de un cifrador por transposición. De esta forma se lograba el objetivo de la confidencialidad, en tanto que la integridad estaba en entredicho y dependía de lo aguerrido y fiel que fuese nuestro mensajero. Si la cinta era robada y se cambiaban los caracteres, podría llegar al receptor un mensaje sin sentido y, lo que es peor, con un duplicado del bastón original podía enviarse un mensaje con sentido completamente distinto al encomendado al mensajero. Haga un viaje mental al pasado e imagínese lo que significaría en aquellos tiempos que el destinatario recibiera el mensaje falso M_F = RENDICIÓN TOTAL en vez del verdadero mensaje M_V = ATACAMOS MAÑANA, ambos de 14 caracteres. Sin duda a más de alguno este desliz le costaría su preciada cabeza.

Un apunte curioso y de cultura general. De estos tiempos tan remotos se debe la famosa frase de ostentar el "bastón de mando" –tan popular en el entorno militar y de nuestros queridos políticos y en particular alcaldes- y que, como es de suponer, en aquella época no se soltaba por ningún motivo puesto que en él residía la seguridad del sistema de información y la vida política de este pueblo de la antigua Grecia.

A pesar de todo esto debe tenerse presente que al reordenar el texto, en el criptograma aparecerán exactamente los mismos caracteres que en el texto en claro y que, por tanto, no se evita en este caso que un intruso detecte fácilmente que nuestro criptosistema es de transposición mediante la simple acción de contabilizar los caracteres del texto cifrado y comparar las frecuencias relativas con las del lenguaje. Esto es, si en el alfabeto de 27 letras la letra E aparece cerca del 13%, la letra A cerca del 10%, etc., no cabe duda que el cifrado ha sido realizado por permutaciones. No obstante, sí se destruyen los digramas, trigramas y, en general poligramas, al separar los caracteres en el texto cifrado.

Ahora bien, aunque se detecte una distribución de caracteres en el criptograma muy parecida a la característica del lenguaje, sólo nos indica eso, que es muy posible que se haya cifrado por transposiciones, pero de nada nos servirá la técnica utilizada en cifradores por sustitución para intentar un criptoanálisis. En este caso, el ataque deberá plantearse con el uso de una técnica denominada anagramación y que consiste en la comparación de bloques de caracteres del criptograma con el objeto de buscar la formación de los poligramas destruidos por el cifrado.

2.5.1. Transposición por grupos

En este tipo de cifra, los caracteres del texto en claro se reordenan por medio de una permutación $\Pi_x(y)$ en donde x indica la acción ejercida sobre el conjunto de caracteres del mensaje M e y es la posición ordenada de los caracteres según la acción x. Luego, si $a_1, a_2, a_3, ..., a_k$ son letras del texto en claro, y Π es una permutación de 1, 2, 3, ... k números, entonces cada carácter C_i del criptograma será el resultado de aplicar dicha permutación sobre ese conjunto de k caracteres. Por ejemplo, sea el conjunto de números [1, 10] = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 y sean x_1 y x_2 dos acciones de permutación tales que x_1 es la acción de ordenar cada grupo de diez caracteres del mensaje de forma que primero envía los caracteres impares y luego los pares y x_2 es la función de ordenar los caracteres del mensaje desde la posición mayor a la menor, entonces se tiene que:

$$\Pi_1 = 1, 3, 5, 7, 9, 2, 4, 6, 8, 10.$$

$$\Pi_2 = 10, 9, 8, 7, 6, 5, 4, 3, 2, 1.$$

Entonces de los grupos indicados, se tiene por ejemplo que:

$$\Pi_1(4) = 7; \Pi_1(9) = 8; \Pi_2(1) = 10; \Pi_2(3) = 8$$

La transposición por grupos será periódica, de período p, tras el cual la permutación aplicada al texto en claro se repite. Esto es, si el mensaje que se desea cifrar $M = m_1m_2m_3 ... m_{10}m_{11}m_{12}$ y la permutación aplicada con período 4 es $\Pi_M = 4132$, entonces el criptograma generado será $C = m_4m_1m_3m_2m_8m_5m_7m_6m_{12}m_9m_{11}m_{10}$.

Ejemplo: Utilizando la permutación $\Pi_M = 24531$ cifre el siguiente mensaje:

M = MANOS ARRIBA, ESTO ES UN ATRACO.

Solución: Aplicando $\Pi_M = 24531$ al texto en claro:

M = MANOS ARRIB AESTO ESUNA TRACO

C = AOSNM RIBRA ETOSA SNAUE RCOAT

Si el período es pequeño, como en el ejemplo anterior, el criptograma podría atacarse fácilmente mediante técnicas de anagramación que se verán más adelante. Una solución a este problema podría consistir en hacer crecer el período de la transposición. En esta línea podría llegarse a la situación límite en que el período es tan largo como el propio mensaje, dando lugar a los denominados cifradores de transposición por series.

2.5.2. Transposición por series

Esta técnica consiste en ordenar el mensaje como una cadena de submensajes, de forma que el mensaje original se transmite como $M' = M_{S1}M_{S2}M_{S3}...$, en donde cada una de las cadenas sigue una función o serie; por ejemplo, M_{S1} puede corresponder a los múltiplos de 3, M_{S2} los números primos superiores a 3, M_{S3} los números pares, etc.

Suponga entonces un mensaje M con un total de 25 caracteres. Si se utilizan las 3 series M_{S1}, M_{S2} y M_{S3} que se indican en ese mismo orden:

M_{S1}: Relación de números primos
M_{S2}: Relación de números pares
M_{S3}: Relación de números impares

entonces la cifra se realizará como sigue:

$M = m_1m_2m_3m_4m_5m_6m_7m_8m_9m_{10}m_{11}m_{12}m_{13}m_{14}m_{15}m_{16}m_{17}m_{18}m_{19}m_{20}m_{21}m_{22}m_{23}m_{24}m_{25}$
$M' = M_{S1}M_{S2}M_{S3}$
$M_{S1} = 1, 2, 3, 5, 7, 11, 13, 17, 19, 23$
$M_{S2} = 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24$
$M_{S3} = 9, 15, 21, 25$
$C = m_1m_2m_3m_5m_7m_{11}m_{13}m_{17}m_{19}m_{23}m_4m_6m_8m_{10}m_{12}m_{14}m_{16}m_{18}m_{20}m_{22}m_{24}m_9m_{15}m_{21}m_{25}$

Al no tener período, este algoritmo de cifrado posee una mayor fortaleza pues dificulta el criptoanálisis, residiendo su seguridad en el secreto de las series utilizadas. No obstante, es necesario recorrer el texto en claro completo por lo que el método es muy lento.

Ejemplo: Utilizando las series $M_{S1}M_{S2}M_{S3}$ vistas anteriormente y en ese orden, cifre el mensaje:

M = ERRAR ES HUMANO, PERDONAR DIVINO.

Solución: El mensaje tiene 27 caracteres. Si se transmite la secuencia M_{S1}, luego M_{S2} y finalmente M_{S3}, se tienen los siguientes bloques:

$M_{S1} = 1,2,3,5,7,11,13,17,19,23$
$M_{S2} = 4,6,8,10,12,14,16,18,20,22,24,26$
$M_{S3} = 9,15,21,25,27$

El mensaje ordenado según las posiciones de los caracteres es:

```
1234567890 1234567890 1234567
ERRARESHUM ANOPERDONA RDIVINO
```

Permutando los caracteres según la serie $M_{S3}M_{S2}M_{S1}$ so obtiene:
C = ERRRS AODNI AEHMN PROAD VNUER IO

2.5.3 Transposición por columnas

Cifrador de transposición por columnas simple

En este tipo de cifrados, se reordenan los caracteres del texto en claro en N_C columnas de forma que el mensaje así escrito se transmite luego por columnas, obteniéndose de esta manera el criptograma. El efecto, al igual que en los demás cifradores por permutación, es desplazar las letras de las posiciones adyacentes.

Por ejemplo, si $N_C = 6$, la columna de cifrados podría quedar como se indica:

Columna de cifrados

C_1	C_2	C_3	C_4	C_5	C_6
C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
C_{13}	C_{14}	C_{15}	C_{16}	C_{17}	C_{18}
C_{19}	C_{20}	C_{21}	C_{22}	...	...

Luego, el criptograma se obtiene leyendo de arriba hacia abajo en las columnas, es decir:

$$C = C_1C_7C_{13}C_{19} \ldots C_2C_8C_{14}C_{20} \ldots C_3C_9C_{15}C_{21} \ldots \ldots C_6C_{12}C_{18} \ldots$$

Para proceder a la función de cifra, primero se busca una cuadrícula en función del tamaño del bloque del mensaje. Si en la búsqueda de dicha cuadrícula quedan espacios en blanco, éstos se rellenan con algún carácter nulo previamente determinado y que conocen el transmisor y el receptor del mensaje, por ejemplo la letra X.

Ejemplo: Cifre el siguiente texto mediante transposición por columnas con $N_C = 6$. Se usará como carácter de relleno la letra X.

M = NUNCA ES TARDE CUANDO LA DICHA ES BUENA.

Solución: Se escribe el texto en columnas como se indica:

N	U	N	C	A	E
S	T	A	R	D	E
C	U	A	N	D	O
L	A	D	I	C	H
A	E	S	B	U	E
N	A	X	X	X	X

Leyendo por columnas el criptograma resultante será:

C = NSCLA NUTUA EANAA DSXCR NIBXA DDCUX EEOHE X.

Para descifrar un criptograma por columnas, el receptor primero calculará el número de filas N_F a partir de la longitud del texto cifrado L_C y el número de columnas N_C, clave secreta que sólo él conoce.

$$N_F = L_C/N_C$$

Hecho esto, escribe el texto cifrado de forma vertical en tantas filas como indique el valor de N_F y procede a leerlo por filas.

Ejemplo: Se ha recibido el criptograma C = PLXIU IEESN GTSOO OEX y se sabe que ha sido cifrado en 6 columnas. Descífrelo.

Solución: Como el criptograma tiene L_C = 18 caracteres y se ha cifrado con N_C = 6, entonces N_F = 18/6 = 3. Se escriben las seis columnas con longitud de tres caracteres, es decir PLX, IUI, EES, NGT, SOO y OEX.

P	I	E	N	S	O
L	U	E	G	O	E
X	I	S	T	O	X

Leyendo por filas y descartando los caracteres de relleno al final de la matriz, se obtiene el mensaje M = PIENSO, LUEGO EXISTO.

En general, para un cifrado por transposición a través de una matriz de dimensiones j x k (j filas y k columnas), existe una relación funcional entre el texto en claro y el criptograma. En el caso de la cifra por columnas, el carácter del texto en claro de la posición iésima se desplaza a la posición $E_t(i)$ debido a la acción de permutación, en donde:

$$E_t(i) = j * [(i-1) \bmod k] + \text{trunc}\,[(i-1)/k] + 1$$

Donde trunc(f(x))es la parte entera de la operación efectuada sobre f(x). Luego, el carácter M_i del texto en claro se representará en el criptograma como:

$$M_i = C_{j*(i-1)\text{mod}k + \text{trunc}[(i-1)/k] + 1}$$

Se aplicará esta ecuación al texto anterior M = PIENSO, LUEGO EXISTO. La posición relativa de los caracteres en el mensaje y en el criptograma será:

```
i:    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
M:    P I E N S O L U E G  O  E  X  I  S  T  O
C:    P L X I U I E E S N  G  T  S  O  O  O  E  X
```

Por ejemplo, se observa que las letras del texto en claro N (i=4), L (i=7), U (i=8), G (i=10), T (i=16) y el digrama EX (i,k=12,13) del texto en claro se desplazan, a las posiciones 10, 2, 5, 11, 12, 17 y 3 respectivamente. Como el receptor conoce el número de columnas N_C empleadas en el cifrado y la longitud L_C del criptograma, deduce que el número de filas N_F = j es igual a L_C/N_C = 18/6 = 3. Comprobación de estos valores en particular:

N: $E_t(4) = 3*[(4\text{-}1) \text{ mod}6] + \text{trunc}[(4\text{-}1)/6] + 1 = 10 \Rightarrow C_{10}$

L: $E_t(7) = 3*[(7\text{-}1) \text{ mod}6] + \text{trunc}[(7\text{-}1)/6] + 1 = 02 \Rightarrow C_2$

U: $E_t(8) = 3*[(8\text{-}1) \text{ mod}6] + \text{trunc}[(8\text{-}1)/6] + 1 = 05 \Rightarrow C_5$

G: $E_t(10) = 3*[(10\text{-}1)\text{mod}6] + \text{trunc}[(10\text{-}1)/6] + 1 = 11 \Rightarrow C_{11}$

T: $E_t(16) = 3*[(16\text{-}1)\text{mod}6] + \text{trunc}[(16\text{-}1)/6] + 1 = 12 \Rightarrow C_{12}$

E: $E_t(12) = 3*[(12\text{-}1)\text{mod}6] + \text{trunc}[(12\text{-}1)/6] + 1 = 17 \Rightarrow C_{17}$

X: $E_t(13) = 3*[(13\text{-}1)\text{mod}6] + \text{trunc}[(13\text{-}1)/6] + 1 = 03 \Rightarrow C_3$

Ejemplo: Cifre en columnas usando una clave $N_C = 3$ el mensaje:

M = LA VIDA ES UNA TÓMBOLA

Solución: Como el texto en claro tiene 18 caracteres y el número de columnas k = 3, entonces el número de filas j = 6. Las posiciones de los caracteres en el criptograma serán:

$$E_t(1)= 6*[(1-1)mod3] + trunc[(1-1)/3] + 1 = 1 \Rightarrow C_1 = M_1 = L$$
$$E_t(2)= 6*[(2-1)mod3] + trunc[(2-1)/3] + 1 = 7 \Rightarrow C_7 = M_2 = A$$
$$E_t(3)= 6*[(3-1)mod3] + trunc[(3-1)/3] + 1= 13 \Rightarrow C_{13} = M_3 = V$$
$$E_t(4)= 6*[(4-1)mod3] + trunc[(4-1)/3] + 1= 2 \Rightarrow C_2 = M_4 = I$$
$$E_t(5)= 6*[(5-1)mod3] + trunc[(5-1)/3] + 1= 8 \Rightarrow C_8 = M_5 = D$$
$$E_t(6)= 6*[(6-1)mod3] + trunc[(6-1)/3] + 1= 14 \Rightarrow C_{14} = M_6 = A$$

Si se continúa con la cifra se obtiene finalmente: C = LIENO OADSA MLVAU TBA.

De igual manera, pueden encontrarse las posiciones que ocupaban los caracteres del texto en claro a partir del criptograma. Por ejemplo los seis primeros caracteres descifrados del mensaje PIENSO LUEGO EXISTO serán:

$$M_1 = C_{3*[(1-1)mod6] + trunc[(1-1)/6] + 1} = C_1 \quad \Rightarrow M_1 = P$$
$$M_2 = C_{3*[(2-1)mod6] + trunc[(2-1)/6] + 1} = C_4 \quad \Rightarrow M_2 = I$$
$$M_3 = C_{3*[(3-1)mod6] + trunc[(3-1)/6] + 1} = C_7 \quad \Rightarrow M_3 = E$$
$$M_4 = C_{3*[(4-1)mod6] + trunc[(4-1)/6] + 1} = C_{10} \quad \Rightarrow M_4 = N$$
$$M_5 = C_{3*[(5-1)mod6] + trunc[(5-1)/6] + 1} = C_{13} \quad \Rightarrow M_5 = S$$
$$M_6 = C_{3*[(6-1)mod6] + trunc[(6-1)/6] + 1} = C_{16} \quad \Rightarrow M_6 = O$$

Ejemplo: Descifre el siguiente criptograma cifrado con $N_C = 4$ columnas. C = CNEAM SAANY IXMNO CNXIT HAOX.

Solución: Como $N_C = 4$ y el criptograma tiene $L_C = 24$ caracteres, por lo que j = 6.

$$M_1 = C_{6*[(1-1)mod4] + trunc[(1-1)/4] + 1} = C_1 \quad \Rightarrow M_1 = C$$
$$M_2 = C_{6*[(2-1)mod4] + trunc[(2-1)/4] + 1} = C_7 \quad \Rightarrow M_2 = A$$
$$M_3 = C_{6*[(3-1)mod4] + trunc[(3-1)/4] + 1} = C_{13} \quad \Rightarrow M_3 = M$$
$$M_4 = C_{6*[(4-1)mod4] + trunc[(4-1)/4] + 1} = C_{19} \quad \Rightarrow M_4 = I$$
$$M_5 = C_{6*[(5-1)mod4] + trunc[(5-1)/4] + 1} = C_2 \quad \Rightarrow M_5 = N$$

Siguiendo con el mismo procedimiento se obtiene el siguiente texto en claro M = CAMINANTE NO HAY CAMINOS.

Como se verá en el próximo apartado, por mucho que con esta operación se destruyan poligramas, mediante una técnica denominada anagramación se podrá atacar el criptograma. Esto es posible

ya que en el cifrado anterior pueden persistir adyacencias de series cortas de letras, por ejemplo digramas característicos, desplazados una distancia constante. Ante ello, existen dos soluciones a este problema: aplicar una doble transposición o bien hacer uso de una clave para permutar las columnas antes de escribir el criptograma.

Cifrador de transposición por columnas simple con clave

Para hacer más difícil el ataque por anagramación, puede utilizarse una clave con el objeto de cambiar la posición relativa de las columnas de la cuadrícula. Esta clave puede ser cualquier combinación de números desde 1 hasta N_C, no obstante se puede asociar una palabra de longitud N_C con todos los caracteres distintos a dicha combinación de números. Por ejemplo si se trabaja con 7 columnas y se desea una permutación de éstas del tipo 2547136, una posible palabra clave sería la palabra PERMISO pues, ordenando los caracteres de dicha clave alfabéticamente, se obtiene precisamente esa permutación: EIMOPRS.

Ejemplo: Cifre por columnas con la clave RELOJ el siguiente mensaje.

M = EL PATIO DE MI CASA ES PARTICULAR, CUANDO LLUEVE SE MOJA COMO LOS DEMÁS

Solución: Escribiendo el mensaje en 5 columnas y luego permutando éstas según la clave RELOJ:

```
R E L O J     E J L O R
E L P A T     L T P A E
I O D E M     O M D E I
I C A S A     C A A S I
E S P A R     S R P A E
T I C U L     I L C U T
A R C U A     R A C U A
N D O L L     D L O L N
U E V E S     E S V E U
E M O J A     M A O J E
C O M O L     O L M O C
O S D E M     S M D E O
A S X X X     S X X X A
```

Escribiendo las columnas resultantes, se tiene:

C = LOCSI RDEMO SSTMA RLALS ALMXP DAPCC OVOMD XAESA UULEJ OEXEI IETAN UECOA.

Si desea provocar una mayor confusión y difusión en el criptograma, en otras palabras rizar el rizo, puede incluir por ejemplo un par de líneas más en la matriz después del fin del mensaje, utilizando las mismas letras del texto. De esta manera, si toma como caracteres de relleno los de las columnas

1ª, 3ª y 5ª del mensaje escrito en columnas antes de aplicar la clave (EPT, IDM, IAA, EPR…), las últimas cuatro filas de la primera matriz serán ahora:

```
. . . . .
O S D E M
A S E P T
I D M I A
A E P R T
```

El criptograma, que se lo dejo como ejercicio, será:

C = LOCSI RDEMO SSDET MARLA LSALM TATPD APCCO VOMDE MPAES AUULE JOEPI REIIE TANUE COAIA.

Cifrador de doble transposición por columnas

Para destruir la adyacencia de series cortas de caracteres que pueden aparecer en una única transposición, también puede utilizarse una segunda permutación. Con ello, el criptograma final se obtiene tras aplicar las siguientes transformaciones:

$$C' = E_1(M)$$
$$C = E_2[E_1(M)]$$

Esto es, se escribe el texto del mensaje en claro M en columnas (operación E_1) en una matriz de dimensiones j' x k' y luego se reordena dicha matriz en otra de dimensión j x k. El efecto de esta doble transposición será separar aún más los caracteres adyacentes y destruir, por tanto, los digramas. Compruebe Ud. mismo que al aplicar una doble permutación se produce una mayor dispersión de los caracteres del texto en claro en el criptograma final. En este caso ya no nos servirá el método de anagramación que será analizado más adelante como herramienta de ataque a la cifra.

2.5.4. Transposición por filas

De forma similar al sistema de cifra por columnas, en esta operación de cifra se escribe el mensaje M en forma vertical, por ejemplo de arriba hacia abajo, con un cierto número de filas N_F que será la clave y luego se lee el criptograma en forma horizontal tal como se indica en el siguiente ejemplo.

Ejemplo: Cifre por transposición de filas con clave $N_F = 3$ el siguiente mensaje:

M = EL PRISIONERO SE ENTREGARÁ EN EL LUGAR YA INDICADO

Solución: Escribiendo el texto verticalmente en tres niveles, se tiene:

```
E R I E S N E R N L A A D A
L I O R E T G A E U R I I D
P S N O E R A E L G Y N C O
```

El criptograma se obtiene escribiendo las tres filas resultantes:

C = ERIES NERNL AADAL IORET GAEUR IIDPS NOERA ELGYN CO

Evidentemente, las operaciones de cifrado y descifrado serán análogas a las vistas en los sistemas de cifra por columnas. Esto es, conocido el número de elementos del criptograma L_C y la clave N_F, se calcula ahora el número de columnas N_C como L_C/N_F y luego se escribe el criptograma de forma horizontal en tantas columnas como sea el valor de N_C encontrado. Leyendo el resultado por columnas, en forma vertical de arriba hacia abajo, se obtiene el texto en claro.

Ejemplo: Descifre el siguiente criptograma de cifra por filas y clave NF = 3.

C = MAPDD ITOOE RURNX.

Solución: Como la longitud del criptograma $L_C = 15$ entonces $N_C = L_C/N_F = 15/3 = 5$. Se escribe el criptograma en cinco columnas y luego se lee de arriba hacia abajo:

M	A	P	D	D
I	T	O	O	E
R	U	R	N	X

Obteniendo el siguiente mensaje M = MIRA TÚ POR DÓNDE

Para hacer las cosas un poco más complicadas, otra forma de cifrar el mensaje, similar a la anterior, es mediante una figura de zigzag de forma que la clave también se encuentra en el nivel de profundidad de dicha figura, como se indica en el próximo ejemplo.

Ejemplo: Utilizando el cifrado por líneas con figura zigzag con una profundidad igual a 3, cifre el mensaje M = EL ESPAÑOL COMO EL JUDÍO, DESPUÉS DE COMER SIENTE FRÍO

Solución: El mensaje se escribe como se indica:

```
E   P   L   O   U   D   U   E   E   E   F
  L S A O C M E J D O E P E D C M R I N E R O
    E   Ñ   O   L   I   S   S   O   S   T   I
```

Luego, leyendo en filas, se obtiene el criptograma:

C = EPLOU DUEEE FLSAO CMEJD OEPED CMRIN EROEÑ OLISS OSTI.

2.5.5. Criptoanálisis de los cifrados por transposición

La técnica de anagramación consistirá en la elección de un conjunto de elementos de una columna o fila, llamado ventana, y su posterior comparación con otras cadenas de caracteres en columnas o filas de igual longitud con el objeto de encontrar digramas comunes que han sido rotos por la transposición. La idea es que dicha ventana recorre todo el texto cifrado y en algún lugar coincidirán todos los digramas con los del mensaje original. A continuación se muestra un caso particular de ataque a una cifra por columnas. Los pasos a seguir ante un cifrado que se sospeche sea de columnas, serán los siguientes:

a) Calcular primero la distribución de frecuencia de los caracteres del criptograma. Si dicha distribución resulta similar a la característica del lenguaje, es muy posible que el criptograma

en cuestión se corresponda con un cifrado por transposición pues el criptograma tiene los mismos elementos que el texto en claro.

b) Se elige una cadena de al menos 7 caracteres del comienzo del criptograma y que se denominará ventana. Con esta ventana se recorrerá el resto del criptograma avanzando en incrementos de un carácter y en cada paso se compararán los digramas que aparecen, fruto de los caracteres de dicha ventana y del resto del texto cifrado. Aunque es recomendable la elección de una ventana grande para poder aplicar con lógica las estadísticas del lenguaje, el tamaño de dicha ventana deberá ser menor que el número de filas que resultase en una cifra por columnas o bien el número de columnas de una cifra por filas.

c) Si la mayoría de los digramas presentan una alta frecuencia, esto indica que puede tratarse de dos columnas de la operación de cifrado. De esta forma, puede reconstruirse la matriz y, por tanto, descifrar el criptograma.

En resumen, la idea es que, dado que tras la operación de cifra se conservan todos los caracteres del texto en claro, eso sí permutados, al comparar un bloque que será parte de una columna con otros bloques, es posible encontrar digramas de alta frecuencia y esto permitirá encontrar el período y, por tanto, romper el cifrado. A continuación se verá cómo funciona este método a través de un ejemplo.

Cifre, por ejemplo, en cuatro columnas el siguiente mensaje M = ESTO NO HAY QUIEN LO ARREGLE.

E	S	T	O
N	O	H	A
Y	Q	U	I
E	N	L	O
A	R	R	E
G	L	E	X

Leyendo por columnas y agrupando en bloques de cinco caracteres el resultado es el criptograma C = ENYEA GSOQN RLTHU LREOA IOEX.

La separación de los digramas del texto en claro dentro del criptograma es, precisamente, el número de filas obtenidas al confeccionar la matriz. Luego, si se elige una ventana, por ejemplo, igual a 4 caracteres ENYE, y se compara con los restantes bloques de 4 caracteres del criptograma, en algún momento se realizará la comparación de la ventana ENYE con los caracteres desplazados un período, es decir la cadena SOQN, obteniéndose en este momento los digramas ES, NO, YQ y EN. Puesto que de estos cuatro digramas ES, NO y EN son muy frecuentes en el lenguaje castellano, podría suponerse que el período de la cifra viene dado por la distancia que hay desde el primer carácter de la ventana hasta el primer carácter de la cadena analizada, en este ejemplo los seis espacios que separan la E de la S en la palabra ESTO del mensaje. Luego, al escribir el criptograma en seis filas, se llega a la matriz anterior que permite encontrar el mensaje original.

Puede generalizarse este método diciendo que se elige una ventana de un tamaño V caracteres, es decir:

$$\text{Ventana} = C_1 C_2 \ldots C_V$$

A continuación, se observan los digramas que se forman al recorrer con esta ventana el resto del texto, es decir:

$$C_1C_{V+1}, C_2C_{V+2}, \ldots, C_VC_{2V}$$
$$C_1C_{V+2}, C_2C_{V+3}, \ldots, C_VC_{2V+1}$$
$$C_1C_{V+3}, C_2C_{V+4}, \ldots, C_VC_{2V+2}$$
$$\ldots\ldots$$

En cada comparación de la ventana con un bloque, se busca la frecuencia relativa de los digramas encontrados de acuerdo a una tabla de digramas de referencia dada y se calculan la media de la muestra y la desviación estándar. Si la media es un valor alto y la desviación estándar es baja, quiere decir que todos los valores de C_aC_b tienen alta probabilidad de ser parte de un texto en claro y que, además, la media alta no es debida solamente a algún digrama aislado de muy alta frecuencia.

Si se dan estas condiciones entonces es probable que el período L del cifrado, las filas de la matriz en el caso de una cifra por columnas, sea igual a la distancia en caracteres que separa a ambas cadenas p_{Cx} y p_{C1}, luego:

$$L = p_{Cx} - p_{C1}$$

donde p_{Cx} es la posición relativa donde comienza la cadena que se está comparando y p_{C1} es la posición de inicio del criptograma y de la ventana. La media se calcula sumando las frecuencias relativas f_r de los digramas en el lenguaje:

$$\overline{X} = \frac{1}{V}\sum_{i=1}^{V} f_r$$

siendo V el tamaño de la ventana en caracteres. La desviación estándar σ será:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{V}(f_r - \overline{X})^2}{V}}$$

Encontrado un período L, puede intentarse extender el tamaño de la ventana hasta dicho valor, con la idea de tener una cadena de caracteres igual a la de una columna en la operación de cifra, o por el contrario buscar L posiciones más adelante otra cadena para ver si también presenta digramas comunes con la que le precede.

Ejemplo: Realice un ataque por anagramación sobre el criptograma que se indica eligiendo una ventana de tamaño V = 4

C = TPNOT OAOPO DRYAD OAURO SUNAS.

Solución: Como V = 4 el bloque será TPNO que se comparará con TOAO, OAOP, AOPO, OPOD, etc. Usando una tabla de digramas se obtiene.

T	t Tt 11	o To 285	a Ta 436
P	o Po 225	a Pa 181	o Po 225
N	a Na 332	o No 222	p Np 49
O	o Oo 40	p Op 131	o Oo 40
Media:	152	205	188
Desviación:	136,77	56,44	161,32

Con la cadena OAOP de la segunda comparación, los digramas to, pa, no y op muestran una media alta asociada con una baja desviación estándar respecto a esa media. Asimismo, en la tercera comparación -cadena AOPO- aparece el digrama ta, de muy alta frecuencia, pero su efecto se enmascara con los digramas np y oo de baja frecuencia, dando una desviación alta. De lo anterior podría deducirse que el período es igual a 5, el tamaño de la cadena más el número de comparaciones hechas antes de dar con el bloque de media alta y desviación baja.

Siguiendo con el método y suponiendo que el período es igual a 5, podría observar los digramas que aparecen desplazando en el criptograma 5 espacios; es decir, comparar por ejemplo la cadena encontrada OAOP con DRYA, luego la cadena DRYA con OAUR y finalmente la cadena OAUR con SUNA, es decir:

od 250	do 354	os 764
ar 493	ra 520	au 72
oy 35	yu 8	un 338
pa 181	ar 493	ra 520

Puesto que se mantiene una media alta, se confirma que el período podría ser igual a 5. No obstante, en estas comparaciones no se cumple que la media alta vaya acompañada de una desviación baja; esto ocurre cuando el tamaño de la ventana es de sólo algunos caracteres como es en este caso y que hace muy difícil y arriesgado aplicar estadísticas tan alegremente. En este ejemplo como la clave $N_C = 5$ y el mensaje tenía muy pocos caracteres se obtienen pocas filas (de hecho $N_F = 5$) y por lo tanto nos ha forzado la elección de una ventana pequeña. Recuerde que éste es un método estadístico, por tanto no infalible, y que para tener un mínimo grado de confianza en los resultados será necesario contar con un criptograma de gran longitud y no pocas veces algo de intuición y suerte. Como ejercicio, descifre Ud. mismo este noble mensaje de una novela de *Alejandro Dumas*, ligeramente modificado.

2.6 De la cifra clásica a los cifradores modernos

El estudio de las técnicas de sustitución-transposición de los apartados anteriores y de los diferentes métodos de criptoanálisis publicados es un buen ejemplo de la evolución de la ciencia de la criptología en la historia. Si fuera necesario establecer una referencia temporal en la que poder separar algoritmos criptográficos simétricos clásicos de los modernos, actuales, sin duda sería en la década de los 40

del siglo XX con la publicación de dos artículos fundamentales que sentarían las bases de la teoría de la información: *A Mathematical Theory of Communication*, en 1948, y *Communication Theory of Secrecy Systems,* en 1949, desarrollados por *Claude Shannon*. Los artículos de *Shannon* propusieron dos técnicas de cifrado en criptosistemas de clave secreta, que resumían los mecanismos anteriores de la historia, a las que llamó difusión y confusión.

Por un lado, la difusión sería la técnica que permitiría dispersar las propiedades estadísticas inherentes al lenguaje en el texto en claro sobre el criptograma, por ejemplo, mediante permutaciones o transposiciones. Por otro lado, la técnica de confusión, permitiría generar confusión, caos, mezcla en el resultado cifrado, de tal forma que la dependencia entre texto en claro, clave y criptograma sería lo más compleja posible e impediría romper el algoritmo (propone aplicar la técnica de sustitución).

Dicho de otra manera, se profundizó en una serie de teorías matemáticas, teoría de números y teoría de la información, que reutilizando el conocimiento previo en técnicas de sustitución y transposición comenzó a definir un nuevo tipo de criptografía. Un tipo de criptografía basada en algoritmos públicos y cuya seguridad se basaba exclusivamente en el conocimiento de la clave criptográfica.

En la criptografía simétrica moderna existen dos tendencias: criptografía simétrica de bloques y criptografía de flujo. En el primer caso, es la evolución de las técnicas clásicas que en lugar de utilizar poligramas para cifrar utilizan bloques de N bits. Por otro lado, la criptografía de flujo es un intento de llevar a la práctica el cifrado de *Vernam* y el esquema OTP. El problema del cifrado de *Vernam* es la distribución de la clave.

Los cifradores de flujo pretende utilizando una clave pequeña conocida exclusivamente por emisor y receptor que exista un generador que en la práctica genere una clave con las propiedades que pudiera encontrarse en un cifrado *Vernam*-OTP. Lógicamente al tratarse de un procedimiento determinista la secuencia clave generada finalmente tendrá un período, pero en la práctica los cifradores de flujo se pueden diseñar para tener periodos enormes, por ejemplo 10^{38} bits. Con la clave generada se realiza una operación or-exclusiva con la información a proteger. En el ejemplo anterior se podría enviar hasta 10^{38} bits (10^{29} Gbit). Si suponemos una conexión de 100 Gbit/s la clave no se repetiría antes de ¡31.709.791.983.764.586.504 años! Estos algoritmos tienen sus ventajas pero también tienen una serie importante de inconvenientes.

No es el objetivo de este apartado profundizar en los detalles de los algoritmos simétricos o de flujo más modernos, de ello hay muy buena información tanto en inglés como en español (consulte la sección de bibliografía). El libro que tiene entre manos no aportaría mucho más en este sentido.

En este apartado se piensa que es más interesante abordar otra cuestión que muchas veces algún profesional de las tecnologías de la información se habrá planteado. ¿Por qué los algoritmos son como son? Comprendido esto, el lector podrá comprender los detalles y nuevos cambios de cualquier algoritmo sin problema.

Los algoritmos de flujo, aunque con múltiples variantes, en esencia consisten en el diseño de polinomios matemáticos (una función a la que damos valores a las variables en función de una serie de bits iniciales que van variando y que se conocen como semilla), polinomios con una serie de propiedades matemáticas que permiten garantizar propiedades estadísticas y de periodicidad del resultado de dicha función, es decir, de los bits de salida que se utilizarán como clave.

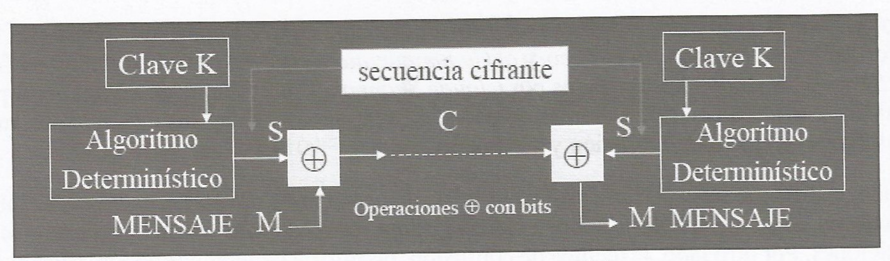


Figura 9. Esquema de un cifrador de flujo.

Comprender el diseño y el porqué de cada elemento en un cifrador simétrico de bloques es diferente. Piénse en lo siguiente: la seguridad de estos algoritmos se denomina seguridad condicional, es decir, a excepción de un ataque por fuerza bruta, en principio intratable (seguridad computacional), el algoritmo es seguro mientras que nadie demuestre lo contrario. Esto es importante no perderlo de vista pues marcará la estructura de los algoritmos de cifrado.

En la criptografía simétrica moderna se intenta maximizar los dos conceptos fundamentales resaltados por *Shannon*, la confusión y la difusión. Mediante el mínimo número de operaciones se debe conseguir minimizar el conocimiento del atacante de las propiedades estadísticas del texto en claro y se debe operar el texto de forma que sea ininteligible e irreversible sin el conocimiento de la clave.

Independientemente del algoritmo que estudie, Triple DES, AES, IDEA, etc., es común la existencia de una serie de criterios para alcanzar difusión y confusión:

1) Operación or-exclusiva.
2) Operación de desplazamiento o rotación.
3) Operaciones de no-linealidad. Por ejemplo, las cajas S de DES o los procedimientos en AES.
4) Estructura de mezcla de las operaciones anteriores.
5) Número de vueltas o repeticiones del conjunto de operaciones.

La operación or-exclusiva tiene una propiedad estadística interesante, además de su sencillez y rapidez de operación, dada una salida existe un 50% de probabilidad que la entrada que lo produce sea un 1 o un 0, esto es muy interesante para conseguir difusión y confusión. Dentro esta tarea, y de nuevo por su sencillez, es común el uso de operaciones de desplazamiento o rotación, es la manera moderna de realizar en la práctica técnicas de transposición de los bits. Todas estas operaciones son importantes pero muchos algoritmos exclusivamente con ellas harían que fuera relativamente sencillo criptoanalizarlos.

En este punto entran en juego operaciones matemáticas de no-linealidad, típicamente reflejadas en una serie de tablas a los que dada una entrada, por sustitución, se produce una salida. En el caso de otros algoritmos con funciones de expansión o reducción como DES se genera una salida que al invertir daría igual a varias entradas posibles. El diseño de estas tablas o caja-S es fundamental en la seguridad de los algoritmos de criptografía simétrica modernos. Existen diferentes formas de crearlas, un manera típica es utilizando funciones de Bent.

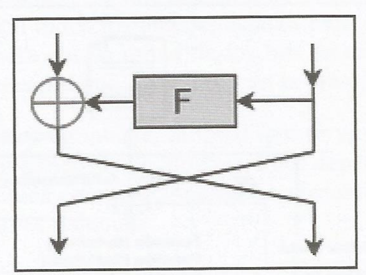
S1 \ COLUMNAS	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
FILAS 0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Figura 10. Una de las 8 cajas S del algoritmo DES. En cada caja entran 6 bit y salen 4.

Y aunque parezca mentira esto es la base de la mayoría de los algoritmos modernos simétricos que trabajan con bloques de bits.

El siguiente paso es definir una estructura lógica que mezcle este tipo de operaciones. Un ejemplo famoso de esto es la red o estructura tipo *Feistel*. La red *Feistel* recibe su nombre en honor a su inventor *Horst Feistel*, famoso por su trabajo en IBM por diseños como el algoritmo LUCIFER en la década de los 70 y porque propuestas suyas derivarían en el algoritmo DES, que se convertiría en estándar en 1976 como el algoritmo criptográfico predominante en las comunicaciones mundiales, al menos hasta 1999.

La red *Feistel* consiste en lo siguiente:



Dado un bloque de N bits, se trocea en dos partes. La parte derecha sale como la nueva parte izquierda y la nueva parte derecha será el resultado de hacer una operación or-exclusiva de la entrada izquierda con una serie de modificaciones, función F, de la entrada derecha.

Por ejemplo, el algoritmo DES utiliza la red *Feistel* y la función F realiza funciones de no-linealidad, desplazamientos, or-exclusivas, etc., para facilitar la confusión y la difusión. Aunque este tipo de estructura se ha utilizado en múltiples algoritmos existen muchas otras opciones, por ejemplo, el actual estándar de cifrado AES no utiliza una estructura *Feistel*. La información a cifrar se maneja en una estructura de datos, matriz de estado, en la cual se le realizan una serie de operaciones según los siguientes esquemas.

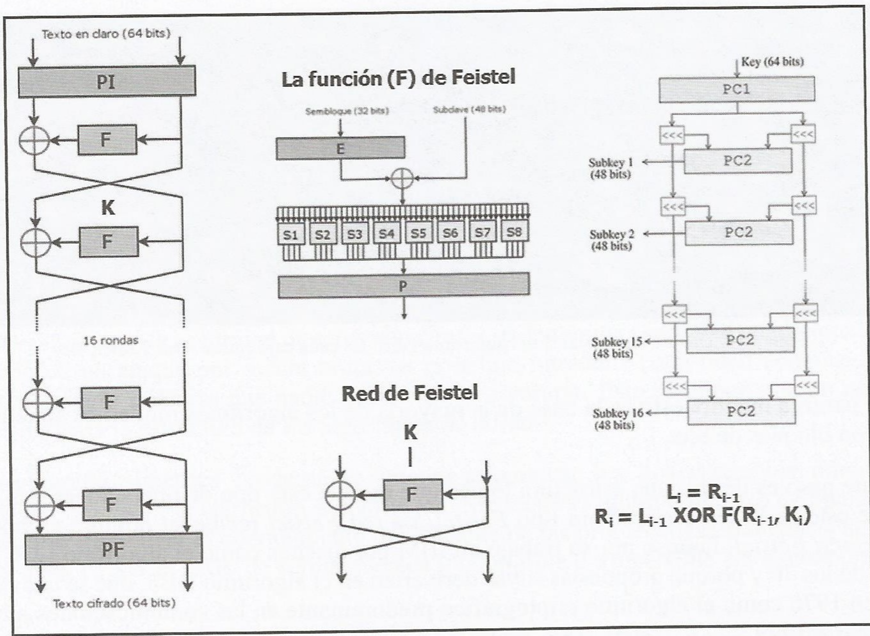


Figura 11. Estructura del algoritmo DES.

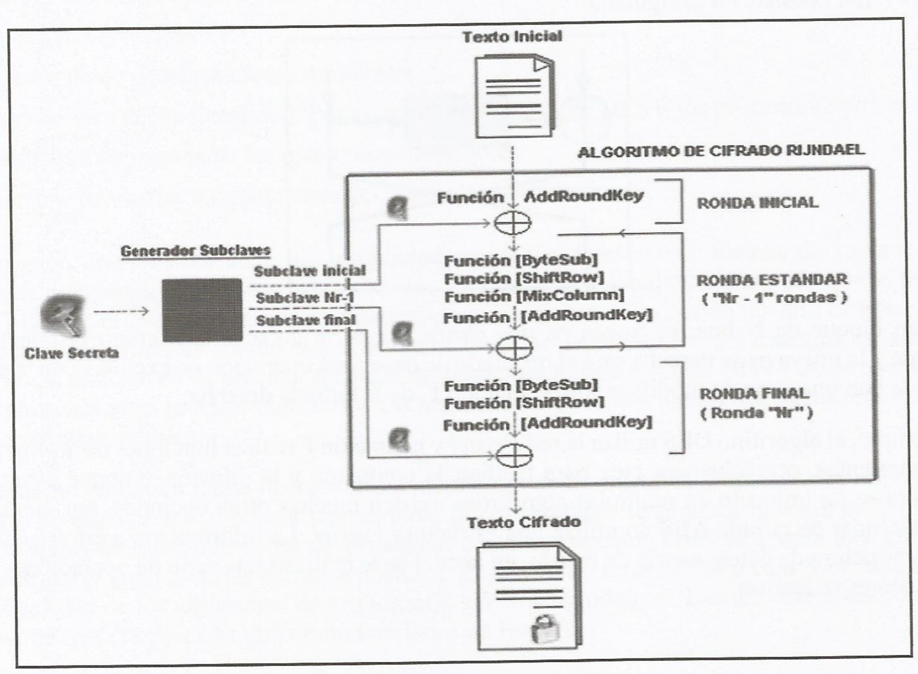


Figura 12. Estructura del algoritmo AES.

En este punto ya se conoce porqué es frecuente el uso de ciertas operaciones y la necesidad de mezclarlas con una red de conexión con una cierta lógica que maximice la difusión y confusión en el criptograma resultante. Pero esto no es suficiente, se debe considerar la condición de seguridad condicional y seguridad computacional que cumplen los algoritmos simétricos de bloques modernos.

Los algoritmos deben maximizar las propiedades de difusión y confusión y deben ser seguros frente a los ataques conocidos (seguridad condicional). Una de las formas que tienen los algoritmos modernos de cumplir las dos condiciones anteriores es repetir la estructura de operaciones seleccionadas un número de veces o vueltas. Por ejemplo, el algoritmo DES repite 16 veces la estructura *Feistel* indicada.

Una pregunta adecuada sería cuál es el número de vueltas adecuadas, esto dependerá del algoritmo concreto. Por ejemplo, el algoritmo AES, el estándar actual, para un tamaño de bloque de 128 bits y clave 128 bits es seguro frente a ataques como criptoanálisis diferencial o criptoanálisis lineal mediante 6 vueltas. Los autores definen el algoritmo con 10 vueltas, 4 más en lo que consideran un posible margen de seguridad.

Como se puede ver es sencillo comprender cómo son las cosas. Más difícil será elegir las operaciones concretas, su orden, cómo relacionarlas y cuantas veces repetirlas considerando aspectos se seguridad, memoria, procesamiento, tamaño del bloque cifrado, etc.

Por ejemplo, el diseño de AES (algoritmo *Rijndael*) hace que dos vueltas del algoritmo produzcan una difusión completa, en el sentido de que, cada bit del estado depende de todos los bits de las 2 vueltas anteriores, es decir, un cambio en un bit del estado es similar a cambiar la mitad de los bits del estado después de dos vueltas. La alta difusión de una vuelta de *Rijndael* es gracias a su estructura uniforme que opera con todos los bits del estado. Para cifradores tipo *Feistel*, como por ejemplo DES, un vuelta sólo opera con la mitad de los bits de estado y una difusión completa se obtiene en el mejor de los casos después de 3 vueltas y en la práctica 4 o más.

Llegados a este punto uno podría preguntarse, al igual que se vio con los criptosistemas clásicos, cómo se criptoanaliza un cifrador simétrico de bloques. Pues depende de muchos factores, recursos matemáticos a su alcance y recursos computacionales, no obstante, al menos públicamente parece que incluso con todo esto el éxito es reducido.

Por ejemplo, uno de los ataques más recientes contra AES[4] para conseguir la clave de cifrado necesita de "sólo" $2^{126.1}$ operaciones para un AES de clave 128 bits, $2^{189.7}$ y $2^{254.4}$ operaciones para AES de 192 y 256 bits respectivamente.

Otra cosa diferente es cómo se implemente estos algoritmos en hardware/software o cómo se relacionen los diferentes bloques cifrados generados. Los cifradores simétricos de bloques, como su nombre indica, trocean el mensaje en claro en diferentes bloques que luego cifrarán. En las últimas décadas se han documentado diferentes formas de cifrar y relacionar los bloques a tratar. Los esquemas clásicos son ECB (*Electronic Code Book*), CBC (*Cipher block chaining*), CFB (*cipher feedback*), OFB (*output feedback*) o CTR (*counter mode*), su uso depende de la información a tratar, si se dispone de toda la información al cifrar o hay que esperar, si se debe transmitir según se recibe

4 Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger (2011). "Biclique Cryptanalysis of the Full AES". *https://en.wikipedia.org/wiki/Advanced_Encryption_Standard#Security*

aunque sólo haya un bloque, la propagación de errores de un bloque cifrado a adyacentes en la transmisión, etc.

Un ejemplo claro de inseguridad es el uso del modo de cifrado ECB, es decir, cifrar bloque a bloque de manera individual. Dependiendo de la información cifrada, es posible manipular o generar situaciones incómodas con este tipo de cifrado. Pero sin duda una de las características que llaman más la atención es la posibilidad de hacer ataques visuales. Imagine el caso que cifra una imagen, los pixeles (bits) son cifrados por bloques. Es factible representar la información cifrada gráficamente y obtener información de la información en claro (la imagen). Piense por ejemplo qué sucedería si se fuerzan cambios visuales bruscos. Suponga que consideramos la información cifrada como si fuera los pixeles de una imagen y forzamos, por ejemplo, que el valor de los bytes sean todo 0s o 1s según una lógica. El resultado puede verse en la siguiente figura.

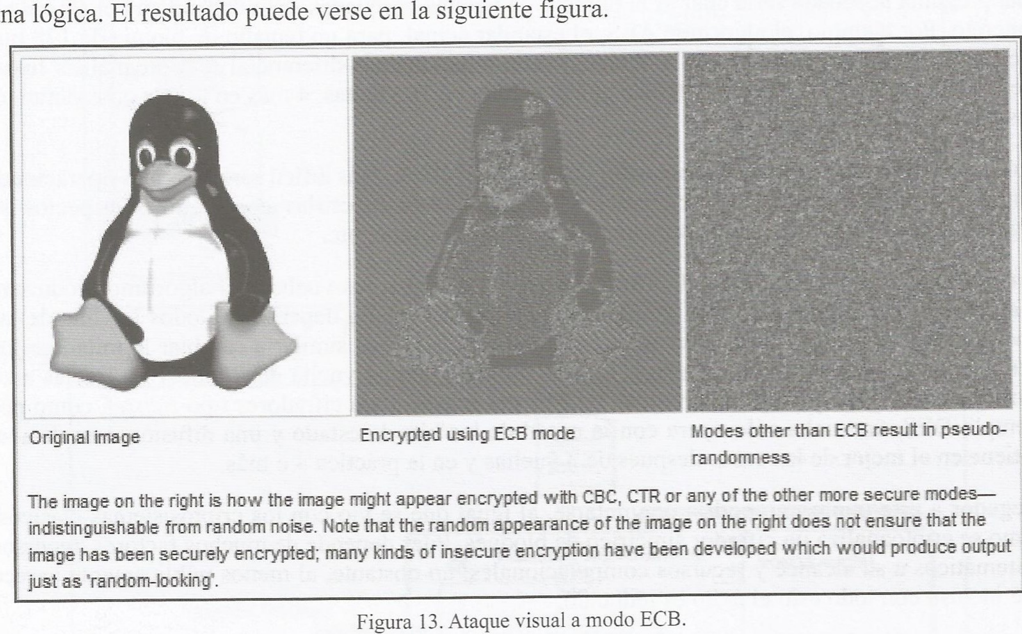


Figura 13. Ataque visual a modo ECB.

Un tema sin duda interesante. Si todo lo reflejado en este apartado ha abierto su curiosidad en profundizar en algoritmos modernos, por favor, le recomendamos consulte la bibliografía reflejada con algunos libros de gran interés.

Capítulo III
Criptografía de clave pública: El algoritmo RSA

La criptografía asimétrica o de clave pública, que tiene sus inicios a finales del año 1976 con el trabajo *News Directions in Cryptography* de *Whitfield Diffie* y *Martin Hellman*, soluciona el problema de distribución de claves existente en la criptografía de clave simétrica. En esta criptografía de clave pública, cada usuario tendrá dos claves, una clave pública y una clave privada y secreta, inversas entre sí de forma que lo que se cifra con una de ellas, se descifra con la otra.

Para comunicarse confidencialmente con otro usuario, se utilizará la clave pública del destinatario, puesto que sólo el destinatario con su clave privada podrá recuperar la información. Si la información intercambiada es una clave de sesión, ésta se podría utilizar para cifrar información con algún cifrado simétrico moderno. Adicionalmente, como sólo el usuario dueño de la clave posee su clave privada, éste podrá cifrar con dicha clave y por tanto firmar digitalmente la información. Esto es así porque cualquiera que conozca su clave pública podrá deshacer el cifrado y por tanto verificar su identidad.

Este es un aspecto muy interesante de la criptografía de clave pública, que según se cifre con la clave pública del destino o la clave privada del emisor, se consigue bien confidencialidad de la información o bien autenticidad del emisor. Y se obtienen por separado, si bien es posible también realizar las dos operaciones de forma simultánea, algo que es imposible en los sistemas de cifra simétrica o de clave secreta, excepto que la clave secreta compartida no esté en entredicho y sólo la conozcan los interlocutores válidos.

Se les denomina sistemas de cifra asimétricos puesto que lo que se cifra con una clave en el extremo emisor deberá descifrarse con la clave inversa en el extremo receptor, es decir claves distintas. Esto a diferencia de los sistemas de cifra con clave secreta que se denominan simétricos porque la misma clave se usa para cifrar y para descifrar.

Como es fácil observar en el mundo real, todas estas características son necesarias en las comunicaciones seguras abiertas en Internet.

Por desgracia, salvo ese hito histórico en 1976 no existen propuestas novedosas que aborden de otra forma el problema de la distribución de claves entre N usuarios. En la década de los 80, los algoritmos basados en curvas elípticas abordaron de otra forma la manera de conseguir algoritmos de criptografía pública. Su principal ventaja consiste en conseguir una seguridad similar a otros algoritmos asimétricos, del tipo exponenciación o similar como RSA, pero con claves de tamaño más pequeño. Esto es muy importante dado que con el avance de las técnicas de criptoanálisis y computación, las claves deben ser cada vez mayores. Si es posible obtener una seguridad similar con un RSA de 4.096 bits que un sistema de cifra de curvas elípticas con una clave de 256 bits, esto presenta importantes ventajas en almacenamiento, transmisión, consumo, etc.

En la historia de la criptografía pública se han documentado multitud de algoritmos asimétricos, entre los más famosos y al cual le dedicaremos el resto del libro, se encuentra el algoritmo RSA, de uso masivo en las comunicaciones en Internet. RSA basa su fortaleza, esto será matizado posteriormente, en la dificultad computacional de factorizar un número compuesto muy grande (del orden de mil bits o superior) resultado del producto de dos primos también grandes, conocido como problema de la factorización entera. Otros algoritmos de cifra asimétrica utilizan principios matemáticos similares, como es el caso del algoritmo propuesto en 1985 por *Taher Elgamal* que basa su fortaleza en la complejidad matemática de resolver logaritmos discretos en un primo grande.

Los siguientes apartados profundizan en el algoritmo RSA por su utilidad práctica en las comunicaciones civiles y comerciales, con el objetivo de convertirse en uno de los resúmenes más completos del tema publicado en lengua española.

3.1. Intercambio de clave de Diffie y Hellman

Los sistemas de cifra con clave pública tuvieron su inicio con la propuesta de *Bailey Whitfield Diffie* y *Martin Hellman* en noviembre del año 1976 para realizar un intercambio de clave computacionalmente seguro en un canal por definición inseguro, usando para ello el problema del logaritmo discreto. Este protocolo de establecimiento de claves entre usuarios que no habían tenido contacto previo, normalmente llamado protocolo DH, fue un hito en la historia de la criptografía y abrió el camino a las comunicaciones seguras en Internet y el comercio electrónico.

El protocolo de intercambio de clave para dos interlocutores es conceptualmente muy sencillo. Es además muy fácil generalizar el protocolo para compartir una misma clave con más de dos interlocutores, permitiendo, por tanto, un intercambio de claves por grupo.

Paso 1:

1) Los usuarios A (Alicia) y B (Bernardo) seleccionan un grupo multiplicativo p (valor que tiene inverso) y un generador $\alpha \in$. Ambos valores son públicos.

Se conoce como generador α dentro de un primo p a aquel número cuyos resultados (restos) de elevar cada uno de los números desde 1 a p-1 módulo p son precisamente todos los resultados (restos) posibles, es decir, los números de 1 a p-1.

Pasos 2 al 6:

2) A genera un número aleatorio a y envía a B: $\alpha^a \bmod p$

3) B genera un número aleatorio b y envía a A: $\alpha^b \bmod p$

4) B calcula $(\alpha^a)^b \bmod p = \alpha^{ab} \bmod p$ y se deshace de b

5) A calcula $(\alpha^b)^a \bmod p = \alpha^{ba} \bmod p$ y se deshace de a

6) El secreto compartido, y por tanto la clave simétrica compartida, entre A y B es el valor $\alpha^{ab} \bmod p$, ya que $\alpha^{ab} \bmod p = \alpha^{ba} \bmod p$.

Suponga que Alicia y Benito desean a intercambiar una clave de sesión dentro del cuerpo primo p = 1.999, con $\alpha = 33$. Por ejemplo el usuario A elegirá su valor privado a 47 y el usuario B elegirá su valor privado b 117.

El algoritmo sigue los siguientes pasos:

1) A calcula α^a mod p = 33^{47} mod 1.999 = 1.343 y se lo envía a B.

2) B calcula α^b mod p = 33^{117} mod 1.999 = 1.991 y se lo envía a A.

3) B recibe 1.343 y calcula 1.343^{117} mod 1.999 = 1.506.

4) A recibe 1.991 y calcula 1.991^{47} mod 1.999 = 1.506.

El valor intercambiado como clave secreta entre Alicia y Bernardo es el número 1.506.

Si un tercero desea atacar y vulnerar este secreto, puesto que conoce los valores de α y p, que son públicos, y puede además interceptar los resultados de las operaciones realizadas por Alicia (α^a mod p) y Bernardo (α^b mod p), tiene dos opciones:

1) Hacer por fuerza bruta todos los cálculos α^x mod p = C con 2 < x < p-1 hasta encontrar el valor C que ha enviado Alicia o Bernardo y así deducir x.

2) Resolver x en α^x mod p = C; es decir resolver el logaritmo discreto x = $\log_\alpha$ C mod p.

Ambas operaciones son computacionalmente inabordables para un primo p grande (sobre los mil bits), de ahí su dificultad o complejidad computacional muy similar a la de la factorización entera que usa RSA.

Aunque este protocolo se convierte en un hito que revoluciona el mundo de la criptografía, no permitía realizar una cifra real de la información, no permitía tampoco la firma digital sobre un documento y, sobre todo, exigía temporalidad en las acciones; es decir, emisor y receptor debían operar el protocolo en tiempo real. Hoy día, este último problema puede solucionarse mediante una variante del protocolo que permite intercambiar un valor secreto generado o conocido a priori y que, de hecho, se usa en aplicaciones de correo electrónico seguro cuando se utiliza *Diffie* y *Hellman* para el intercambio de clave.

Este protocolo puede verse de la siguiente forma:

1) Alicia selecciona un primo p_A, un generador α_A, una clave privada (a) y calcula una clave pública $\alpha_A{}^a$ mod p_A, siendo además público p_A y α_A.

2) Bernardo selecciona un primo p_B, un generador α_B, una clave privada (b) y calcula una clave pública $\alpha_B{}^b$ mod p_B, siendo además público p_B y α_B.

3) Alicia calcula una clave de sesión K = ($\alpha_B{}^b$ mod p_B)^a mod p_B = $\alpha_B{}^{ba}$ mod p_B.

4) Este valor K lo conoce Alicia antes de iniciar el protocolo con Bernardo.

5) Alicia envía a Bernardo el siguiente valor: $\alpha_B{}^a$ mod p_B.

6) Bernardo recupera la clave de sesión ($\alpha_B{}^a$ mod p_B)^b mod p_B = $\alpha_B{}^{ba}$ mod p_B.

7) No es necesario que ambos desarrollen el protocolo simultáneamente.

Alicia desea enviar una clave secreta K a Bernardo mediante el protocolo anterior. Los datos de los dos interlocutores son:

Alicia: p_A = 5.849; α_A = 211; a = 31; $\alpha_A{}^a$ mod p_A = 3.726.

Bernardo: $p_B = 6.673$; $\alpha_B = 135$; $b = 88$; $\alpha_B{}^b \bmod p_B = 5.933$.

Nota: el cálculo y comprobación de los generadores puede hacerlo con el software ExpoCrip que se indica en el Apéndice. Software Educativo. Los cálculos del protocolo son los siguientes:

a) Alicia calcula clave $K = (\alpha_B{}^b \bmod p_B)^a \bmod p_B = 5.933^{31} \bmod 6.673 = 5.706$

b) Alicia envía a Bernardo $\alpha_B{}^a \bmod p_B = 135^{31} \bmod 6.673 = 1.376$

c) Bernardo recupera la clave $(\alpha_B{}^a \bmod p_B)^b \bmod p_B = 1.376^{88} \bmod 6.673 = 5.706$

3.2. Principios del algoritmo RSA

La propuesta de *Diffie* y *Hellman* de 1976 tenía como inconveniente que no permitía realizar una cifra real de la información, no permitía la firma digital sobre un documento y, sobre todo, exigía temporalidad en las acciones, es decir, emisor y receptor debían operar el protocolo en tiempo real[1].

En febrero de 1978, es decir poco más de un año después de aquel intercambio de clave de *Diffie* y *Hellman*, otros tres investigadores, *Ron Rivest*, *Adi Shamir* y *Leonard Adleman*, proponen un sistema de cifra asimétrico que llevará las iniciales de sus apellidos. El algoritmo se patenta como RSA.

A diferencia del intercambio de clave de DH, que basaba su fortaleza en la dificultad computacional de calcular logaritmos discretos en primos muy grandes, RSA basa su fortaleza, en teoría, en la dificultad computacional de factorizar un número compuesto muy grande, producto de dos primos grandes. Ambos problemas tienen una complejidad algorítmica similar y son inabordables computacionalmente si se toman una serie de consideraciones.

No fueron fáciles los primeros años de este algoritmo pues nadie creía en su utilidad, y con el paso de los años existirían muchos otros algoritmos entre los que elegir para realizar funciones similares. No obstante, su seguridad y su versatilidad dio la razón a sus inventores y finalmente se convirtió en estándar, PKCS #1 RSA *Cryptography Standard* y es utilizado masivamente en las comunicaciones hoy día en Internet.

Sin embargo, aunque *Rivest*, *Shamir* y *Adleman* son los autores de RSA, el mismo algoritmo de cifra asimétrico basado en la dificultad de factorizar números grandes fue descubierto mucho antes, o eso afirma el gobierno británico. En el año 1969 el *Government Communications Headquarter*s (GCHQ) en Gran Bretaña comienza a trabajar en la idea de poder distribuir claves a través de una cifra no simétrica, llegando en el año 1973, cinco años antes, a la misma conclusión que los creadores de RSA. Esa investigación fue dirigida por el matemático *Clifford Cocks*. Desgraciadamente el trabajo fue considerado como secreto por el gobierno británico por lo que su contenido no se hizo público ni se patenta como invento. Hoy día pueden consultarse estos trabajos desde la propia web del GCHQ.

---

1 El algoritmo RSA no tiene el problema de temporalidad en el envío de una clave porque si deseamos enviar al receptor R un número secreto N, como se verá más adelante simplemente hacemos el cálculo $N^{eR} \bmod n_R = C$ (donde e_R y n_R son la clave pública del receptor) y le enviamos el criptograma C en el cual se encontrará enmascarado el secreto N. El receptor podrá recuperar el número secreto N cuando lo desee usando su clave privada y secreta d_R, sin restricción temporal respecto al envío del emisor.

Independientemente de la casuística de la historia de la criptografía pública para el intercambio de claves, está claro que no se puede entender la seguridad hoy día en Internet sin el algoritmo RSA.

Conceptualmente este algoritmo es muy sencillo y puede reflejarse en dos expresiones matemáticas:

$$\text{Cifrado } C = M^e \bmod n$$

$$\text{Descifrado } M = C^d \bmod n$$

Donde M es el mensaje, C el criptograma, e la clave pública del destino, d la clave privada del destino y n es el cuerpo de cifra o módulo público del destino.

La operación mod es una operación clásica en aritmética modular y consiste simplemente en calcular el resto de dividir M elevado a un número e entre un número n (M^e/n) y el resto de dividir C elevado a un número d entre un número n (C^d/n).

La configuración adecuada de los números e, d y n, permite recuperar un mensaje M de un mensaje cifrado C. Precisamente estos números e, d y n constituyen las claves criptográficas utilizadas en el algoritmo. RSA es un algoritmo asimétrico y como tal utiliza un par de claves para conseguir sus objetivos: una clave pública formada por los números (e, n) y una clave privada formada por los números (d, n). En función de si se cifra con la clave pública o con la clave privada se obtendrá cifrado de la información (confidencialidad) o firmado de la información (integridad y autenticidad).

Si el mensaje a proteger fuera una clave de sesión K el propio algoritmo RSA facilita por su diseño el intercambio de claves de manera segura en un canal inseguro. Si un usuario A desea enviar una clave K a B realizaría $C = K^{eB} \bmod n_B$, B y sólo B obtendría la clave K calculando $C^{dB} \bmod n_B$.

En los siguientes apartados se analizará en detalle cada uno de estos aspectos para comprender perfectamente cómo utilizar de manera segura este algoritmo, destacando aspectos poco conocidos como el concepto de claves parejas o la existencia de mensajes no cifrables.

A continuación se va a describir la forma adecuada de generar claves en el algoritmo RSA de forma que pueda ser utilizado en el mundo real con unas mínimas garantías de seguridad para proporcionar confidencialidad, integridad y autenticidad en la información a proteger o intercambiar.

En adelante, si algún concepto de matemática discreta no le queda claro, por favor consulte el Apéndice. Fundamentos de Matemáticas Discretas donde encontrará mayor información.

3.3. Generación de claves para el algoritmo RSA

El algoritmo RSA trabaja con dos claves, una clave pública (e, n) y una clave privada (d, n). Los valores e, d y n tienen las siguientes particularidades:

$n = p \times q$

El valor del número n es un producto de dos números primos. Actualmente los primos son de valor igual o superior a 512 bits. Los números primos se mantienen en secreto.

$\phi(n) = (p - 1)(q - 1)$

Cada usuario calculará el indicador de *Euler* ϕ de ese módulo n. En el caso de dos primos se demuestra que el indicador se calcularía como $\phi(n) = (p - 1)(q - 1)$. Ese valor va a ser su secreto o trampa, un número que sólo conoce su dueño.

$1 < e < \phi(n)$ y mcd $[e, \phi(n)] = 1$

Cada usuario elegirá un valor de e dentro el siguiente intervalo $1 < e < \phi(n)$ y para asegurarse que exista el inverso multiplicativo, y por tanto la clave privada d, debe cumplirse que mcd $[e, \phi(n)] = 1$. El valor e y el módulo n forman la clave pública.

$d = \text{inv}\ (e, \phi(n))$

Usando ahora el Algoritmo Extendido de *Euclides* se calcula el valor d que debe permanecer secreto. El valor d y el módulo n forman la clave privada.

Por tanto, el protocolo detallado para la generación de un par de claves RSA y su intercambio entre dos usuarios Alicia (A) y Bernardo (B), es de la siguiente forma:

1) Los usuarios Alicia y Bernardo eligen cada uno un grupo $n = p * q$. Actualmente p y q son primos de valores iguales o superiores a 512 bits.

2) Alicia y Bernardo hacen público el cuerpo o módulo de trabajo n.

3) En el caso de Alicia ese módulo será $n_A = p_A * q_A$ y en el caso de Bernardo será $n_B = p_B * q_B$.

4) Los valores de los primos p y q serán un secreto, sólo conocido por los propietarios de esas claves.

5) Cada usuario calculará el indicador de *Euler* ϕ de ese módulo n, que en este caso de dos primos es $\phi(n) = (p - 1)(q - 1)$. Así, Alicia calcula $\phi_A = (p_A - 1)(q_A - 1)$ y Bernardo calcula $\phi_B = (p_B - 1)(q_B - 1)$. Ese valor va a ser su secreto o trampa, un número que sólo conoce su dueño.

6) A partir de ese valor trampa ϕ se calculará ahora la clave pública e y la correspondiente clave privada d.

7) Cada usuario elegirá un valor de clave pública e dentro el siguiente intervalo: $1 < e < \phi(n)$. Para asegurarse que exista el inverso multiplicativo, y por tanto la clave privada d, debe cumplirse que mcd $[e, \phi(n)] = 1$.

8) Ese valor e será la segunda parte de su clave pública, además del módulo n.

9) Alicia elige como clave pública $1 < e_A < \phi_A$ y Bernardo elige como clave pública $1 < e_B < \phi_B$.

10) Usando ahora el Algoritmo Extendido de *Euclides*, cada usuario calcula su clave privada d. Alicia calcula $d_A = \text{inv}\ (e_A, \phi_A)$ y Bernardo calcula $d_B = \text{inv}\ (e_B, \phi_B)$.

11) Luego, cada usuario tiene dos valores que forman su clave pública, n y e, y un valor secreto que es su clave privada d.

12) Además del valor d, también guardarán en secreto p y q, que le servirán para acelerar la operación de descifrado utilizando el teorema chino del resto como se verá posteriormente.

Recuerde que hacer públicos los valores de e y n, no pone en peligro la clave privada d puesto que para calcular $d = \text{inv}\ [e, \phi(n)]$ hace falta conocer la trampa $(p - 1)(q - 1)$, es decir los primos p y q.

Como se analizará en detalle en el capítulo de seguridad del algoritmo, si la longitud de los primos es adecuada esta tarea se convierte en computacionalmente intratable. Es común en 2012, debido al avance en la reducción del tiempo de computación en factorización del RSA, trabajar con módulos n superiores a 2000 bits. De hecho, si accede al enlace seguro de un banco online, lo más probable es que se encuentre con un certificado digital X.509 que muestre una clave pública RSA cuyo módulo es de 2.048 bits.

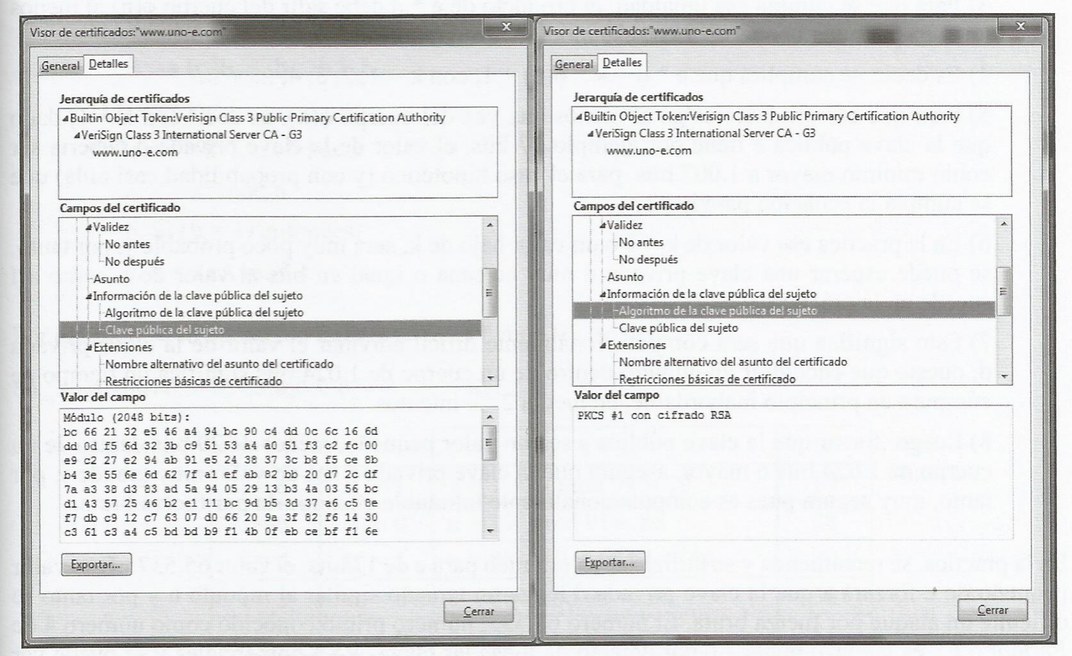


Figura 14. Certificado digital banco UNO-E (fecha acceso 29/12/2012).

Seguramente después de generar de forma automática varias claves reales, por ejemplo de 1.024 ó 2.048 bits con el software genRSA que se presenta en el Apéndice. Software educativo, se preguntará porqué la clave pública e que propone el programa está entre 10 y 15 bits y, en cambio, la clave privada d que se calcula tiene siempre un valor muy cercano al tamaño del módulo n. En los próximos apartados saldrá de dudas y verá que dicha clave pública es, además, un valor estándar igual e impuesto para todos.

3.3.1. Diseño y elección de claves RSA: valores de p, q y e

De lo descrito en el apartado anterior pudiera dar la sensación que las restricciones a la hora de elegir los valores p, q y e (indirectamente d) son pequeñas y se reducen a cumplir unos rangos y tamaños mínimos, pero ni mucho menos se trata de eso. Se indican a continuación los criterios a considerar.

Recomendaciones en la elección del valor e

Es recomendable utilizar un valor de clave pública e relativamente pequeño comparado con el módulo n. Los motivos siguen el siguiente razonamiento:

1) La trampa $\phi(n)$ será igual a (p - 1)(q - 1) y como p y q son primos de al menos 512 bits, el tamaño de $\phi(n)$ será entonces aproximadamente igual al de n, ligeramente menor pero con igual número de bits.

2) Como la clave pública e y la clave privada d son inversas en el cuerpo $\phi(n)$, es decir d = inv [e, $\phi(n)$], entonces se cumple la siguiente relación: e * d mod $\phi(n)$ = 1.

3) Para que se cumpla esa igualdad, el producto de e * d debe salir del cuerpo $\phi(n)$ al menos una vez para que la operación en ese módulo nos devuelva el valor 1.

4) Es decir, se cumplirá que e * d = k * $\phi(n)$ + 1, con k = 1, 2, 3, 4, ...

5) Para que ese producto salga al menos una vez del cuerpo $\phi(n)$, es decir con k = 1, dado que la clave pública e tiene por ejemplo 17 bits, el valor de la clave privada d debería ser como mínimo mayor a 1.007 bits, para el caso hipotético (y con probabilidad casi nula) que se cumpla la ecuación para k = 1.

6) En la práctica ese valor de k = 1 o un valor bajo de k, será muy poco probable y, por tanto, se puede esperar una clave privada d muy cercana o igual en bits al valor de n como así sucede en la práctica.

7) Esto significa que será computacionalmente difícil adivinar el valor de la clave privada d, puesto que encontrar un número dentro de un cuerpo de 1.024 bits significa un tiempo de cómputo en principio inabordable, en media $2^{1.023}$ intentos.

8) Luego, forzar que la clave pública e sea un valor pequeño, menor de 20 bits dentro de un cuerpo de 1.024 bits o mayor, asegura que la clave privada d sea un valor muy grande y, por tanto, muy segura pues es computacionalmente intratable encontrarla por fuerza bruta.

En la práctica, se recomienda y se utiliza un valor único para e de 17 bits, el valor 65.537[2]. Este valor pequeño de e forzará a que la clave privada d tenga un tamaño similar al módulo n y por tanto se dificulte un ataque por fuerza bruta. El número 65.537, número primo conocido como número 4 de *Fermat* o F_4, es la clave pública e por defecto de todas las claves RSA comerciales y se puede ver en los certificados digitales; lo que será distinto y propio en cada clave son los primos p y q, y por lo tanto el módulo n.

Adicionalmente a las ventajas anteriores, existe un aspecto significativo del número 4 de *Fermat* a destacar. La codificación binaria de este número sólo tiene dos bits iguales a 1 (65.537 = 10000000000000001b = 010001h) lo cual tiene una importante utilidad en la eficiencia de cómputo.

Tradicionalmente los sistemas de cifra asimétrica o de clave pública son mucho más lentos que los sistemas de cifra simétrica, en una relación que oscila entre cien y mil veces más lentos. Este es el principal motivo por el cual se suele recomendar su uso exclusivamente para el intercambio de claves (el cifrado de una pequeña información, una clave simétrica de sesión por ejemplo de centenas de bits) o para el firmado digital (el cifrado de un hash criptográfico también de centenas de bits). Si bien es cierto que estas restricciones dependen mucho del algoritmo asimétrico y del contexto donde se utilice.

Limitando el escenario en el que la tasa de cifra sea crítica, es muy importante valorar la eficiencia del proceso cifrado y descifrado. Dado que el número e = F_4 tiene muchos ceros, al utilizarlo como

2 Es posible elegir otros valores de e de tamaño parecido o más pequeño. Algunos se han demostrado que no son seguros.

exponente (N^e mod n) la operación de cifrado se realiza de forma muy rápida. Esto se puede entender fácilmente siguiendo el algoritmo de exponenciación rápida que se explica a continuación.

Algoritmo de exponenciación rápida

Si se desea realizar la operación A^B mod n:

1) Obtener la representación binaria del exponente B de k bits:

$B_2 \rightarrow b_{k-1}b_{k-2}...b_i... b_2b_1b_0$

2) Hacer x = 1

3) Para i = k-1, ..., 0 hacer

a. $x = x^2$ mod n

b. Si ($b_i = 1$) entonces

c. x = x∗A mod n

Por ejemplo, si la operación a realizar es 19^{83} mod 91, se tiene:

$83_{10} = 1010011_2 = b_6b_5b_4b_3b_2b_1b_0$

		x = 1	x
i = 6	$b_6 = 1$	$x = 1^2*19$ mod 91 = 19	19
i = 5	$b_5 = 0$	$x = 19^2$ mod 91= 88	88
i = 4	$b_4 = 1$	$x = 88^2*19$ mod 91 = 80	80
i = 3	$b_3 = 0$	$x = 80^2$ mod 91 = 30	30
i = 2	$b_2 = 0$	$x = 30^2$ mod 91 = 81	81
i = 1	$b_1 = 1$	$x = 81^2*19$ mod 91 = 80	80
i = 0	$b_0 = 1$	$x = 80^2*19$ mod 91 = 24	24

Efectivamente, 19^{83} mod 91 = 24.

Como se aprecia en la tabla anterior, además de la operación elevar al cuadrado, sólo se multiplica por la base cuando se tiene un bit uno en el exponente. Por tanto, para el exponente 1010011 en el primer paso del algoritmo no se hace operación alguna, porque el resultado es la misma base A con la que se inicia el cálculo, después existen 6 operaciones de elevar al cuadrado, que requieren poco tiempo de máquina, y sólo tres multiplicaciones. Si el exponente en cuestión tiene muchos ceros como en el caso de F_4 entonces la operación más realizada del algoritmo es la de elevar el valor de x al cuadrado que es rápida.

Por tanto, en la práctica la operación de cifrado con la clave pública se realiza mucho más rápido que la operación de descifrado. Eso tiene una utilidad práctica en las comunicaciones actuales basadas en el clásico modelo de comunicación cliente-servidor. Imagínese la comunicación entre un cliente (navegador Web) y un servidor Web mediante el protocolo https (un acceso tradicional a la página

Web de un banco) en el que se requiere la autenticación de la página del banco, esto es que la página sea la que dice ser.

1) El servidor se identifica mediante su clave pública de 2.048 bits.

2) La operación que debe hacer el cliente para enviar la clave de sesión simétrica, por ejemplo de 128 bits a un servidor, una vez conocida su clave pública mediante un certificado digital, es muy rápida. La operación sería por ejemplo $(128 \text{ bits})^{(17 \text{ bits})}$ mod (2.048 bits) y dicho exponente de 17 bits, que es lo más importante aquí, con 15 bits en cero.

3) Por el contrario, la operación de descifrado que deberá realizar el servidor una vez recibido el criptograma será más pesada pues se asemejará a lo siguiente: $(2.048 \text{ bits})^{(2.048 \text{ bits})}$ mod (2.048 bits).

4) En teoría los terminales con menos capacidad de computación (el cliente) tienen un coste menor al utilizar la criptografía mientras que los servidores (con mayor capacidad de computación) tienen un mayor coste computacional.

En resumen, el valor universal para la clave pública de e es F_4, pero en el algoritmo RSA se podría usar en principio cualquier valor de e que estuviese comprendido entre 3 y $\phi(n)$ - 2. Si una clave pública e válida, es decir que tiene su inverso en $\phi(n)$, fuese un valor muy alto y cercano al cuerpo de cifra n, se podría obtener una clave privada extremadamente pequeña cuyo ataque por fuerza bruta sería trivial. Adicionalmente existen otras razones por las que no se debe dejar al usuario la generación del número e de forma aleatoria dentro del rango permitido y que se abordarán en un próximo apartado.

Recomendación en la elección de primos p y q

Una de las fortalezas del algoritmo RSA reside en la imposibilidad de obtener la clave privada d a partir de la clave pública e. Esta imposibilidad se debe a que no es factible que partiendo del módulo n puedan obtenerse los primos p y q, salvo que sea posible factorizar en un tiempo razonable ese número n tan grande. Si se conocen p y q, puede calcularse $\phi(n) = (p-1)(q-1)$ y por tanto d = inv [e, $\phi(n)$]. Para que la afirmación anterior sea cierta, los primos a elegir deberán tener una serie de características para que no sea computacionalmente abordable factorizar n, recuperar p y q de n.

Imagínese, por ejemplo, el caso trivial de p = q y n = p * q, entonces $n = p^2 = q^2$, y encontraría rápidamente sus valores simplemente calculando la raíz cuadrada de n. Esta tarea tiene para n bits una complejidad algorítmica equivalente a la multiplicación de dos números de igual cantidad de bits, por lo cual no debería jamás usarse. Además de la grave debilidad de poder encontrar fácilmente la raíz cuadrada de n y con ello el valor del primo p que se ha elevado al cuadrado, se obtiene un número p de claves privadas parejas (verá su significado más adelante). Si el valor de p es de 512 bits, se tendría un valor altísimo ($2^{512} = 1{,}3 \times 10^{154}$) de claves privadas parejas, lo que no es en absoluto recomendable. Se concluye, por tanto, que los primos p y q deben elegirse como números grandes y distintos. Así mismo, se recomienda que ambos primos estén separados algunos bits, aunque luego esto en la práctica no se tome en cuenta, incluso en software estándar como OpenSSL como se comprobará más adelante.

Además de estas características que deberían cumplir los primos p y q, en la literatura se recomendaba el uso primos seguros y primos fuertes para generar claves RSA.

Un primo q se dice que es seguro si, además de ser primo, es el resultado de multiplicar por dos un primo p menor y sumarle uno. Por ejemplo el número 23 es primo seguro porque 23 = 2 * 11 + 1, siendo 11 y 23 primos. Para una mayor profundización en este tema, puede consultar la Tesis Doctoral del Dr. *Raúl Durán* o consultar en la red por primos fuertes y seguros. En la sección de bibliografía recomendada encontrará información interesante para ello.

El uso de primos seguros y/o primos fuertes en la actualidad no afecta a la calidad de la clave RSA debido a los nuevos algoritmos de factorización, si bien existe alguna discrepancia pues el mayor tiempo de búsqueda de tales primos no compensa los posibles beneficios.

3.3.2. Clave privadas y públicas parejas

Cuando se genera una clave RSA se usa como trampa el indicador de *Euler* $\phi(n)$ para calcular la clave privada d a partir del conocimiento de la clave pública e. Puesto que mcd $[e, \phi(n)] = 1$, se asegura que el inverso d existe y que, además, es el único inverso de la clave pública e en ese cuerpo $\phi(n)$.

No obstante, como es lógico, la cifra se hace posteriormente en el cuerpo público n para que todo el mundo pueda utilizarlo. Y en dicho cuerpo n ya no se cumple que el único inverso de la clave pública e sea la clave privada d. De hecho, hay al menos otro valor diferente a d que cumple las mismas funciones y que, por tanto, permite descifrar lo cifrado con la clave pública.

Claves privadas parejas

A las claves que cumplen con lo indicado en el párrafo anterior se les llama Claves Privadas Parejas o de forma abreviada CPP. Esto es algo en principio inesperado porque normalmente siempre se ha dicho que un sistema de cifra asimétrico, como lo es RSA, tiene una única clave pública y, por lo tanto, también una única clave privada. Para RSA esto ha resultado ser falso.

Se explicará con un sencillo ejemplo. Sea una clave RSA con p = 37, q = 41, n = 1.517, $\phi(n)$ = (p-1)(q-1) = 1.440, e = 13, d = 997. Se cifrará el valor 1001 con la clave pública e = 13 y luego se descifrará el criptograma con la clave privada d = 997. Como son números no muy grandes, puede comprobar estos cálculos haciendo uso, por ejemplo, de la calculadora científica de Windows o la de su sistema operativo favorito.

Cifrado: 1.001^{13} mod 1.517 = 1.088

Descifrado: 1.088^{997} mod 1.517 = 1.001. Se ha recuperado el secreto.

Pero si se usan los números 277, 637 y 1.357 como si fuesen la clave privada d, se obtendría:

Descifrado con d' = 277: 1.088^{277} mod 1.517 = 1.001

Descifrado con d' = 637: 1.088^{637} mod 1.517 = 1.001

Descifrado con d' = 1.357: $1.088^{1.357}$ mod 1.517 = 1.001

Lo que ha sucedido en el ejemplo anterior es que en el cuerpo de cifra n = 1.517 con una clave pública e = 13, los valores 277, 637 y 1.357 son claves privadas parejas (CPP) d' que cumplen la misma función que la clave privada d = 997.

Si crea esta clave con el software *genRSA* que encontrará explicado en el "Apéndice C. Software educativo", se obtiene lo que observa en la figura.

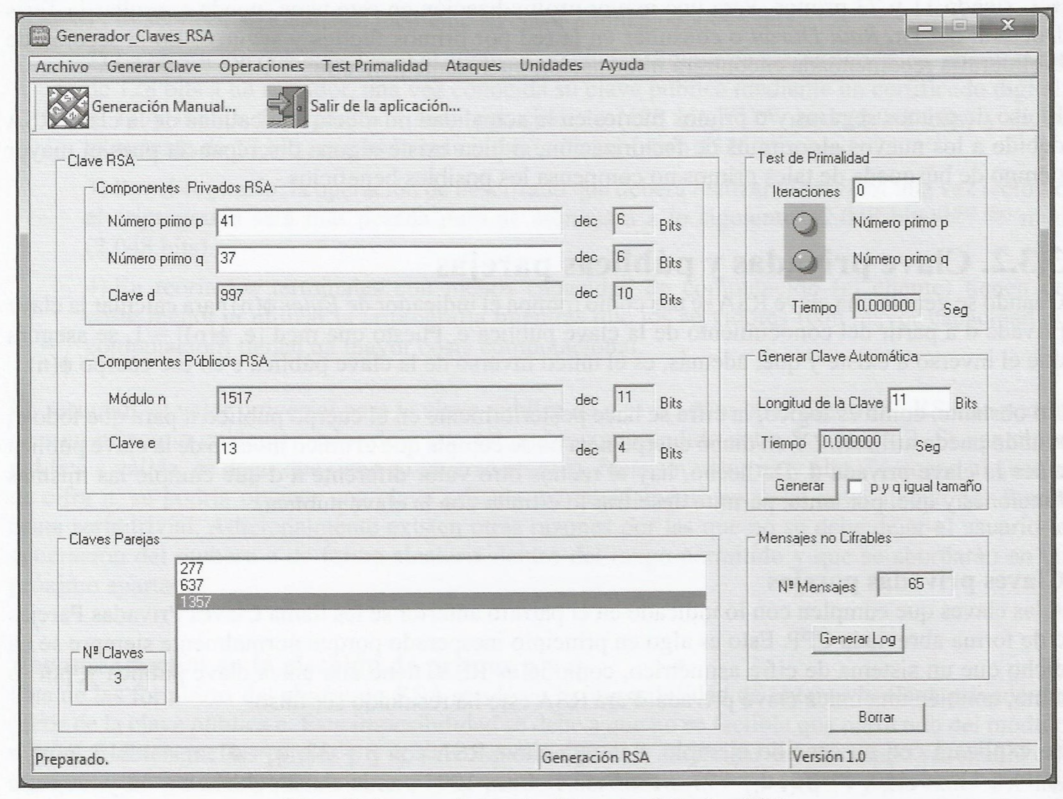


Figura 15. Las tres claves privadas parejas de la clave RSA.

Toda clave RSA tendrá como mínimo una clave privada pareja. Como más adelante podrá comprobar, la cantidad de claves privadas parejas dependerá fuertemente de los primos p y q, siendo las ecuaciones que nos entregan este valor de CPP las siguientes.

Si $\gamma = \text{mcm}\ [(p-1), (q-1)]$ y sea $d_\gamma = e^{-1} \bmod \gamma = \text{inv}\ (e, \gamma)$

La clave privada d tendrá λ claves parejas d_i de la forma:

$$d_i = d_\gamma + i\,\gamma \qquad 1 < d_i < n$$
$$i = 0, 1, \ldots \lambda \qquad \lambda = (n - d_\gamma)/\gamma$$

Sea p = 13 y q = 19, entonces n = 247, $\phi(n) = 216$. Si elige e = 41, la clave privada será d = inv (41, 216) = 137.

Cálculo de las claves privadas parejas:

1) $\gamma = \text{mcm}\ [(p-1),(q-1)] = \text{mcm}\ (12, 18) = 36$

2) $d_\gamma = inv(41, 36) = 29$

3) $d_i = d_\gamma + i\,\gamma = 29 + i*36$

4) $d_i = 29, 65, 101, 137, 173, 209, 245.$

5) $\lambda = \lfloor (n - d_\gamma)/\gamma \rfloor = \lfloor (247 - 29)/36) \rfloor = 6{,}05 = 6$

Observe que aparece el valor 137 que es la clave privada d y puede comprobar que la separación entre todas ellas es $d_\gamma = 36$.

Genere una clave RSA con p = 211, q = 239 y e = 131. Compruebe que se obtienen las siguientes 13 claves privadas parejas y entre ellas se encuentra la clave privada: [3.461; 7.031; 10.601; 14.171; 17.741; 21.311; 24.881; 28.451; 32.021; 35.591; 39.161; 42.731; 46.301; 49.871].

Si el módulo de la clave es muy pequeño será fácil atacarla por fuerza bruta. A primera vista, esto no es algo muy agradable pero no hay de qué preocuparse porque para los valores estándar que se usan en la práctica con cuerpos de cifra de al menos mil bits sucederá todo lo contrario. Como las claves privadas parejas dependerán fuertemente de la elección de los primos p y q. Para obtener una clave con la mínima cantidad de claves privadas parejas, es decir solamente una, se recomienda por ejemplo usar primos seguros.

Es mejor verlo con un ejemplo. Si genera estas 4 claves RSA, en donde cambia sólo ligeramente el valor del primo q, encontrará grandes variaciones en la cantidad de claves privadas parejas.

<table>
<tr><th colspan="2">Clave 1: p = 53, q = 59, e = 11, d = 1371. CPP: 2879 (una)</th></tr>
<tr><td>Clave 2: p = 53, q = 61, e = 11, d = 851
CPP: 71, 1.631, 2.411, 3.191 (cuatro)

Clave 3: p = 53, q = 73, e = 11, d = 2.723
CPP: 851, 1.787, 3.659 (tres)</td><td>Clave 4: p = 53, q = 79, e = 11, d = 1.475
CPP: 71, 227, 383, 539, 695, 851, 1.007, 1.163, 1.319, 1.631, 1.787, 1.943, 2.099, 2.255, 2.411, 2.567, 2.723, 2.879, 3.035, 3.191, 3.347, 3.503, 3.659, 3.815, 3.971, 4.127 (veintiséis)</td></tr>
</table>

De los números primos usados en el ejemplo anterior, se extrae una primera conclusión. El primo p = 53 no es un primo seguro porque se obtiene como (26*2 + 1) y 26 obviamente no es primo. Todos los primos q, excepto el primero q = 59 = 29*2 + 1, con 29 primo, tampoco son primos seguros porque: 61 = (30*2 + 1), 73 = (36*2 + 1), 79 = (39*2 + 1) y los números 30, 36, 39 no son primos.

Además, si en vez de usar sólo uno de los primos como primo seguro o bien primo fuerte se hace con los dos, la clave que resulte tendrá siempre una única clave privada pareja, el valor mínimo.

Al usar primos seguros, en la inmensa mayoría de casos se obtendrán claves RSA con una sola clave privada pareja. En muy pocos casos esto no se cumple. Por ejemplo, si los primos seguros son p = 23 y q = 83, para la clave pública e = 31 se obtiene una sola CPP, en cambio si la clave pública es e = 19, se obtienen 2 claves privadas parejas. Sí se asegura, no obstante, que se tendrá un número bajo muy cercano a la unidad de claves privadas parejas.

Queda claro que la mejor solución para obtener un número mínimo de CPP, es decir una, pasa por usar primos seguros o fuertes, aunque de forma menos automatizada todavía es posible generar claves óptimas con un sola CPP y 9 números no cifrables (se verá más adelante) sin usar primos

seguros, por ejemplo con p = 59.771 y q = 50.159 como muestra la figura, si bien 59.771 cumple con alguna condición de primo fuerte.

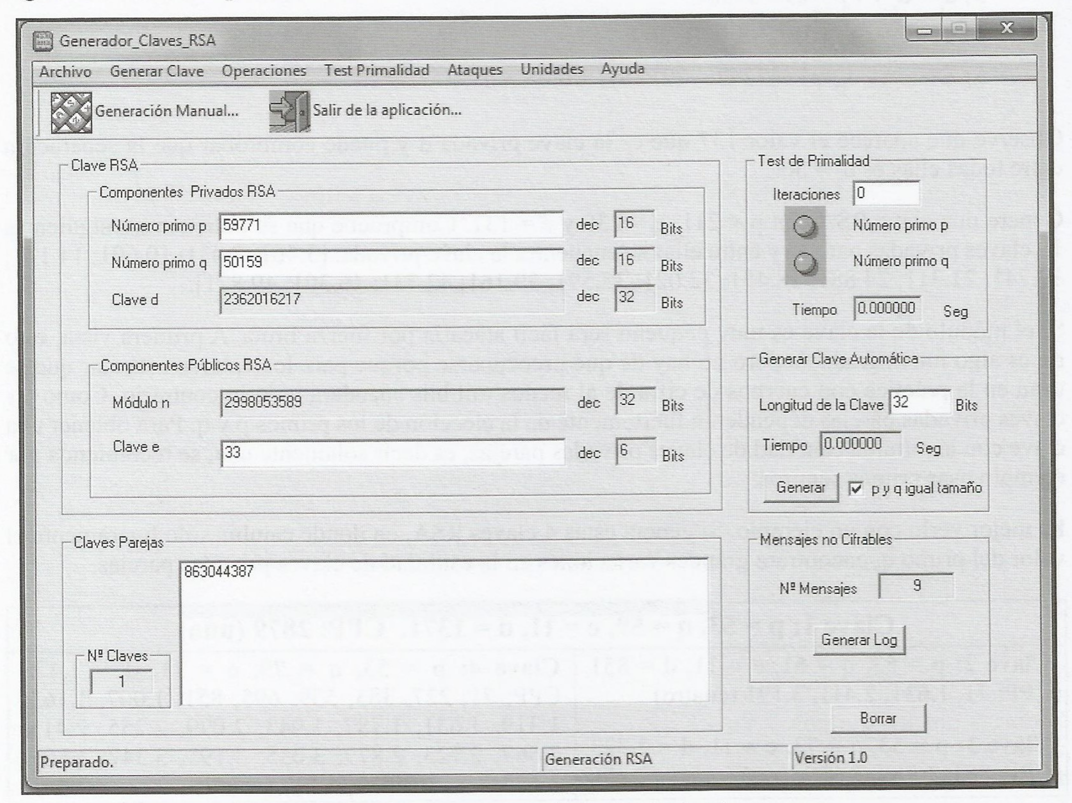


Figura 16. Clave RSA con una sola CPP generada sin primos seguros.

Independientemente de todo lo mencionado, la recomendación que se hace en el estándar ANSI X9.31 es usar primeros fuertes (*strong primes*) en vez de primos seguros (*safe primes*). Puede comprobar con el software genRSA que al generar claves con primos fuertes también se obtiene una única clave privada pareja.

Claves públicas parejas

Al igual que se habla de claves privadas parejas, también existen las claves públicas parejas. Las ecuaciones son iguales a las vistas para CPP salvo que en vez de usar el valor de la clave privada d, se usa el valor e de la clave pública, y viceversa. Si una determinada clave RSA tiene, por ejemplo, 6 claves privadas parejas, tendrá también 6 claves públicas parejas como se muestra a continuación.

Sea nuevamente p = 13 y q = 19, entonces n = 247, ϕ(n) = 216. Si elige e = 41, la clave privada será d = inv (41, 216) = 137.

Cálculo de las claves públicas parejas:

1) γ = mcm [(p-1),(q-1)] = mcm (12, 18) = 36

2) e_γ = inv (d, γ) = inv (137, 36) = inv (29, 36) = 5

3) $e_i = e_\gamma + i\,\gamma = 5 + i*36$

4) e_i = 5, 41, 77, 113, 149, 185, 221.

5) $\lambda = (n - e_\gamma)/\gamma = (247 - 5)/36) = 6,72 = 6$

Por ejemplo si con la clave anterior se cifra (firma) el valor 24 usando d = 137, ésta se podrá comprobar con la clave pública e o bien con cualquiera de sus claves públicas parejas como se muestra a continuación firmando el valor 24.

Firma con clave privada: 24^{137} mod 247 = 215

Comprobación de firma con clave pública: 215^{41} mod 247 = 24

Comprobación de firma con claves públicas parejas:

215^{5} mod 247 = 24
215^{77} mod 247 = 24
215^{113} mod 247 = 24
215^{149} mod 247 = 24
215^{185} mod 247 = 24
215^{221} mod 247 = 24

A primera vista, no resulta agradable el conocimiento de estas claves parejas en RSA ya que realizan la misma función que una clave privada o una clave pública, más aún si se tiene en mente lo que se dice habitualmente de este sistema de cifra asimétrico: que existe una única clave pública (e, n) y una única clave privada (d, n), y que lo que se cifra con una de ellas se descifra con la otra. De hecho, todos los certificados digitales X.509 que trabajan con RSA (la inmensa mayoría) tienen este tipo de claves. La figura 17 muestra una clave de 1.024 bits con más de 3.000 claves privadas parejas generada con genRSA.

La pregunta importante es si este tipo de claves parejas son un problema real. En la actualidad las herramientas reales que utilicen primos de al menos 512 bits y clave pública el número 4 de *Fermat*, generarán un valor mayor o menor de claves privadas parejas pero sus valores serán todos muy próximos al valor del módulo n y, por lo tanto, será intratable adivinarlos o intentar romperlos por fuerza bruta. Recuerde que dada un clave RSA por un tercero, no se sabrá cuántas claves privadas parejas tiene ni tampoco sus valores. Este es un motivo adicional para que exista esa gran diferencia de tamaños entre la clave pública e y la clave privada d.

¿Qué pasará ahora con las claves públicas parejas? La situación en este caso es ligeramente diferente en tanto los valores de las claves públicas parejas podrán tener valores muy pequeños y, además, la clave es por definición pública y no tiene ningún secreto. No obstante, el hecho de que pueda comprobarse una firma digital usando un valor distinto a la clave pública oficial no deja de ser extraño, paradójico e incluso preocupante

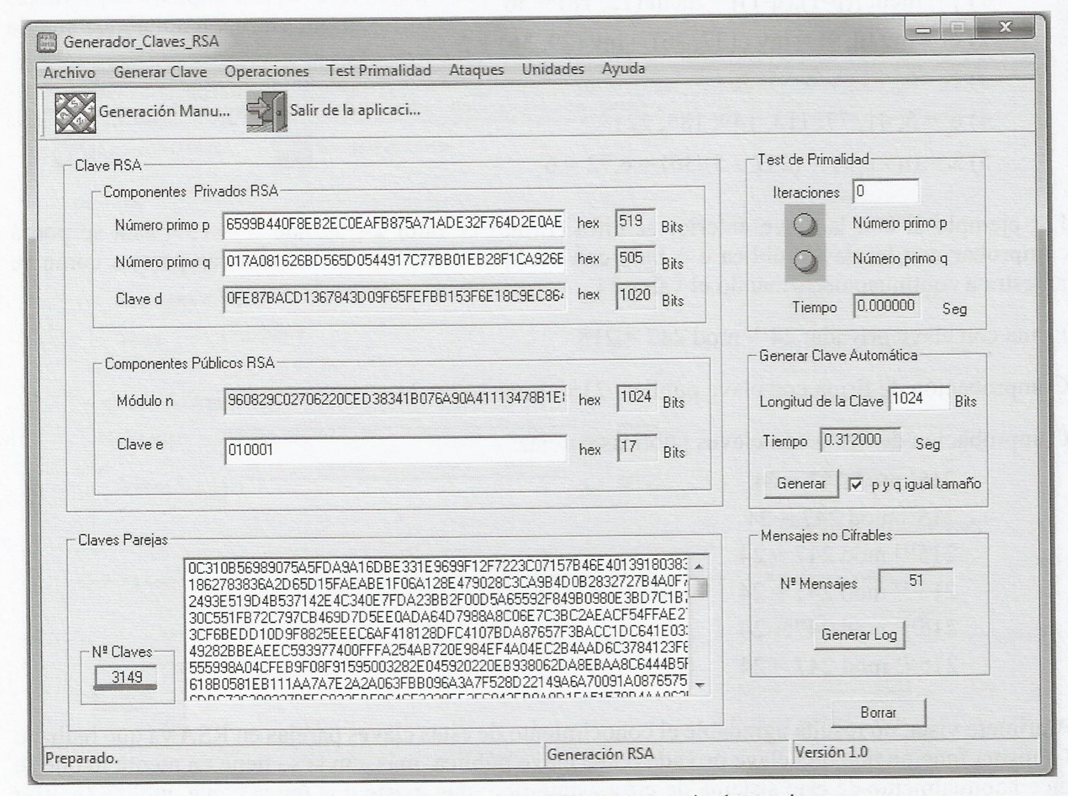


Figura 17. Clave RSA con más de 3.000 claves privadas parejas.

3.4. Cifrado y descifrado de información y mensajes

En apartados anteriores ya se ha abordado brevemente cómo funciona el algoritmo RSA para cifrar y descifrar información, es decir, para garantizar la confidencialidad de una información ya que nadie que no posea la clave privada d podrá recuperar la información protegida.

Matemáticamente las operaciones de cifrado y descifrado se describen como:

$$M^e \bmod n \mid C^d \bmod n$$

Siendo (e, n) la clave pública y (d, n) la clave privada. La clave pública se utilizará para cifrar la información y la clave privada para descifrarla usando el mismo cuerpo de cifra. Es sencillo comprender cómo funciona la operación de cifrado con RSA para obtener confidencialidad. He aquí un ejemplo en que Alicia desea enviar un mensaje secreto M a Bernardo.

1) Alicia conoce la clave pública de Bernardo, n_B y e_B, la cual utiliza para enviarle un mensaje de manera confidencial.

2) Alicia calcula y envía a Bernardo el siguiente criptograma $C = M^{eB} \bmod n_B$.

3) Dado que Bernardo es el único que conoce su clave privada d_B, sólo él podrá realizar la siguiente operación: $M = C^{dB} \bmod n_B$. Es decir, $M = C^{dB} \bmod n_B = [M^{eB} \bmod n_B]^{dB} \bmod n_B = [M^{eB}]^{dB} \bmod n_B$. Puesto que en la operación $[M^{eB}]^{dB} \bmod n_B$ los exponentes se anulan entre sí porque son inversos, en recepción Bernardo recuperará el mensaje M enviado por Alicia dentro del cuerpo n_B.

En realidad, lo que sucede es lo siguiente:

1) Como $d_B = \text{inv}\ [e_B,\ \phi(n_B)]$, entonces $e_B * d_B = k * \phi(n_B) + 1$.

2) Por tanto $[M^{eB}]^{dB} \bmod n_B = M^{k * \phi(nB) + 1} \bmod n_B = M \times M^{k * \phi(nB)} \bmod n_B$.

3) Por el teorema de *Euler* se tiene que $M^{k * \phi(n)} \bmod n = 1$.

4) Luego $M * M^{k\,\phi(nB)} \bmod n_B = M * 1 = M$ y se recupera el mensaje.

Una vez comprendidas estas cuestiones, sería interesante destacar cómo se puede operar en la práctica con un mensaje M de cualquier tamaño para cifrarlo de manera adecuada, ya que hasta ahora en todos los ejemplos se consideraba que el mensaje era un número, puesto que ese es el uso que se hace en la práctica de RSA. En este caso el mensaje M deberá ser troceado y codificado de forma que el tamaño de cada trozo sea menor en bits que el tamaño de n, pues sólo deben cifrarse elementos del cuerpo n.

Aunque múltiples codificaciones serían posibles, una codificación tradicional es trocear el mensaje en bloques de bytes tomando como entrada el valor decimal del código ASCII. Se seleccionan bloques de tamaño un byte menos que el tamaño en bytes del módulo. Por ejemplo, si el módulo n de RSA es el número 39.618.947, que en binario es 10010111001000100110000011, con 26 bits, se debe cifrar bloques de 3 bytes, es decir 24 bits, pues así todos los valores posibles a cifrar serán elementos de n.

Si el mensaje es M = CALCULANDO BLOQUES, se obtendría 6 bloques:

CAL, CUL, AND, O_B, LOQ, UES.

Antes de poder cifrar esos bloques, es necesario obtener una codificación numérica de los mismos. Consultando por ejemplo la tabla ASCII, sería posible asignar una codificación al bloque juntando la codificación binaria de cada carácter del bloque.

CAL = 01000011 01000001 01001100	$M_1 = 4.407.628$
CUL = 01000011 01010101 01001100	$M_2 = 4.412.748$
AND = 01000001 01001110 01000100	$M_3 = 4.279.876$
O_B = 01001111 00100000 01000010	$M_4 = 5.185.602$
LOQ = 01001100 01001111 01010001	$M_5 = 5.001.041$
UES = 01010101 01000101 01010011	$M_6 = 5.588.307$

Una vez obtenida la codificación ya es posible cifrar y descifrar como se ha descrito anteriormente. Así el cifrado y descifrado para los seis bloques será como se indica más abajo.

En este caso ya no podrá usar la calculadora científica de Windows (quizás sí otras calculadoras de otros sistemas operativos) pues para números grandes le entregará el siguiente mensaje de error "*invalid input for function*". Se recomienda usar el software denominado Fortaleza de Cifrados que se presenta en el Apéndice. Software educativo. De la misma manera, se recomienda tener precaución al usar software para cálculos modulares que puede encontrar en Internet y en aplicaciones para teléfonos móviles pues la mayoría de ellos entregan resultados erróneos en una operación tan simple como A^B mod n cuando los números son grandes.

Bloque 1: $C_1 = 4.407.628^{31}$ mod 39.618.947 = 6.734.280
$M_1 = 6.734.280^{5.110.495}$ mod 39.618.947 = 4.407.628

Bloque 2: C_2: $4.412.748^{31}$ mod 39.618.947 = 21.898.080
M_2: $21.898.080^{5.110.495}$ mod 39.618.947 = 4.412.748

Bloque 3: C_3: $4.279.876^{31}$ mod 39.618.947 = 33.215.194
M_3: $33.215.194^{5.110.495}$ mod 39.618.947 = 4.279.876

Bloque 4: C_4: $5.185.602^{31}$ mod 39.618.947 = 30.956.224
M_4: $30.956.224^{5.110.495}$ mod 39.618.947 = 5.185.602

Bloque 5: C_5: $5.001.041^{31}$ mod 39.618.947 = 8.150.418
M_5: $8.150.418^{5.110.495}$ mod 39.618.947 = 5.001.041

Bloque 6: C_6: $5.588.307^{31}$ mod 39.618.947 = 22.796.302
M_6: $22.796.302^{5.110.495}$ mod 39.618.947 = 5.588.307

Como se puede observar, se ha recuperado el mensaje M = CALCULANDO BLOQUES. Para recuperar la información original simplemente se pasaría a binario el número recuperado en el descifrado y se forman bloques de 8 bits leyendo de derecha a izquierda. En la última lectura, si es necesario se añaden ceros a la izquierda hasta formar 8 bits.

3.4.1. Mensajes no cifrables

A lo largo de la historia de la criptografía múltiples algoritmos, simétricos y asimétricos, han presentado pequeñas debilidades que deben ser sorteadas. Algunas de ellas son, por ejemplo, la existencia de claves no recomendadas que o bien no cifran la información a proteger o lo hacen de una manera predecible. Por ejemplo, en el algoritmo criptográfico simétrico DES se detectaba la presencia de claves débiles y semidébiles, que no cumplían con el principio de cifra única planteado por *Claude Shannon*, entregando soluciones conocidas como soluciones falsas. Estos temas relacionados con la Teoría de la Información puede verlos en la sección de apéndices.

Una idea similar sucede en el algoritmo RSA con la existencia de mensajes no cifrables, o mejor dicho, números no cifrables, en adelante NNC.

Esto es muy fácil de entender. Suponga que se desea cifrar un número N (N^{clave} mod n), con una clave que puede ser la clave pública e de destino si lo que se busca es confidencialidad, o bien la clave privada d del emisor si lo que se busca es la integridad y autenticidad. El número N podrá tener cualquier valor desde 0 hasta n-1, siendo n el módulo. De todos los valores posibles de N existirán algunos que, aunque se cifren, irán en claro.

Así, dos casos obvios de números que se cifran y no obstante se envían en claro serán el 0 y 1 puesto que:

$$0^{clave} \bmod n = 0$$
$$1^{clave} \bmod n = 1$$

Otro valor no tan obvio pero que puede demostrarse fácilmente que se transmite en claro es el número n-1 puesto que:

$$n\text{-}1^{clave} \bmod n = n\text{-}1$$

Por ejemplo para la clave RSA con n = 323 y e = 11, se obtiene (puede comprobarlo con la calculadora de su sistema operativo favorito):

$$0^{11} \bmod 323 = 0$$
$$1^{11} \bmod 323 = 1$$
$$322^{11} \bmod 323 = 322$$

Además de esos tres valores, en RSA existirán siempre como mínimo otros 6 valores que irán en claro. Por tanto, cualquier clave RSA tendrá como mínimo 9 números no cifrables.

En el ejemplo anterior, los otros seis valores que se transmiten en claro son:

$$18^{11} \bmod 323 = 18$$
$$152^{11} \bmod 323 = 152$$
$$153^{11} \bmod 323 = 153$$
$$170^{11} \bmod 323 = 170$$
$$171^{11} \bmod 323 = 171$$
$$305^{11} \bmod 323 = 305$$

Pero, ¿cómo pueden encontrarse estos valores de números no cifrables? A continuación encontrará la solución.

A diferencia de las claves privadas y públicas parejas, cuyos valores se obtenían de forma inmediata mediante una ecuación en la que existía, además, una separación constante entre dichas claves, el cálculo de los números no cifrables NNC es más laborioso porque habrá que hacer un ataque de cifrado por fuerza bruta en el espacio de los primos p y q para comprobar qué valores de X nos entregan las siguientes igualdades:

$$X^e \bmod p = X \text{ para } 1 < X < p\text{-}1$$
$$X^e \bmod q = X \text{ para } 1 < X < q\text{-}1$$

Como es de esperar, para claves reales de 1.024 o más bits, resulta computacionalmente intratable realizar esos cálculos dentro de los primos p y q de al menos 512 bits cada uno. Por lo tanto, para

estas claves reales excepto los valores 0, 1 y n-1, no será posible encontrar los demás números no cifrables. No obstante, no tiene porqué preocuparse, siempre es posible comprobar que el mensaje cifrado es diferente del mensaje original, evitando transmitirlo en claro.

Las ecuaciones que intervienen en el cálculo de estos números no cifrables son las siguientes:

La cantidad de números no cifrables σ dentro de un cuerpo n será:

$$\sigma_n = [1 + \text{mcd}(e\text{-}1, p\text{-}1)][1 + \text{mcd}(e\text{-}1, q\text{-}1)]$$

Los números no cifrables serán:

$$N = [q\{\text{inv}(q, p)\}N_p + p\{\text{inv}(p, q)\}N_q] \bmod n$$

con:

N_p las soluciones de $N^e \bmod p = N$

N_q las soluciones de $N^e \bmod q = N$

Como se puede apreciar, la única dificultad de cálculo se encuentra en las dos últimas ecuaciones, que significan aplicar fuerza bruta con todos los valores de N candidatos a ser número no cifrable, siendo 1 < N < p-1 para el primo p y 1 < N < q-1 para el primo q.

Por ejemplo, sea p = 13; q = 17; n = p*q = 221; e = 7; d = inv (7, 192) = 55. Luego la cantidad de números no cifrables σ_n será:

$$\sigma_n = [1 + \text{mcd}(e\text{-}1, p\text{-}1)][1 + \text{mcd}(e\text{-}1, q\text{-}1)]$$

$$\sigma_{221} = [1 + \text{mcd}(6, 12)][1 + \text{mcd}(6, 16)] = (1+6)(1+2) = 21$$

Los números no cifrables serán:

NNC = [q{inv (q, p)}Np + p{inv (p, q)}Nq] mod n

NNC = [17{inv (17, 13)}Np + 13{inv (13, 17)}Nq] mod 221

Como inv (17, 13) = inv (4, 13) = 10

inv (13, 17) = 4

NNC = [{17*10}Np + {13*4}Nq] mod 221

NNC = [170*Np + 52*Nq] mod 221

Por fuerza bruta se encuentran las soluciones de $Np = N^7 \bmod 13$

Np = {0, 1, 3, 4, 9, 10, 12}

Por fuerza bruta se encuentran ahora las soluciones de $Nq = N^7 \bmod 17$

Nq = {0, 1, 16}

Por tanto:

NNC = [170*{0, 1, 3, 4, 9, 10, 12} + 52*{0, 1, 16}] mod 221

NNC = [{0, 170, 510, 680, 1.530, 1.700, 2.040} +{0, 52, 832}] mod 221

NNC = [0+0, 0+52, 0+832, 170+0, 170+52, 170+832, ..., 2.040+832] mod 221

NNC = [0,52,832,170,222,1.002,510,562,1.342,680,732, 1.512,1.530,1.582,2.362,1.700,1.752,2.531,2.040,2.092,2.872] mod 221

Reduciendo cada uno de esos veintiún números módulo 221:

NNC = [0, 52, 169, 170, 1, 118, 68, 120, 16, 17, 69, 186, 204, 35, 152, 153, 205, 101, 51, 103, 220]

Y ordenando de menor a mayor, aparecen los 21 números de la clave RSA no cifrables:

NNC = [0, 1, 16, 17, 35, 51, 52, 68, 69, 101, 103, 118, 120, 152, 153, 169, 170, 186, 204, 205, 220]

Observe, como curiosidad, que a excepción del valor 0, los números de los extremos siempre suman el valor del módulo 221: 1 + 220; 16 + 205; ... 101 + 120; 103 + 118.

Minimizando los números no cifrables

Siguiendo la ecuación indicada en el apartado anterior y dado que e-1 será siempre un número par, puede concluirse que la cantidad de números no cifrables será mínima cuando:

mcd (e - 1, p - 1) = 2
mcd (e - 1, q - 1) = 2

Entonces:

σ_n = [1 + mcd (e-1, p-1)][1 + mcd (e-1, q-1)] = [1 + 2][1 + 2] = 9

Una clave RSA que tiene como máximo 9 números no cifrables y 1 clave privada pareja se conoce como clave óptima. Una manera sencilla de lograr con bastante probabilidad este mínimo de NNC, consiste en elegir p y q como primos seguros.

Suponga que p = 2r + 1 y q = 2s + 1, donde r, s, p y q son primos grandes.

Como e-1 es par, con estos primos seguros se obtendrán los siguientes valores del Máximo Común Divisor:

mcd (e - 1, p - 1) = mcd (e - 1, (2r + 1) - 1) = mcd (e - 1, 2r) = 2 o bien 2r.
mcd (e - 1, q - 1) = mcd (e - 1, (2s + 1) - 1) = mcd (e - 1, 2s) = 2 o bien 2s.

Por lo tanto, los cuatro valores posibles de números no cifrables para p y q primos seguros serán:

$\sigma_n = (1 + 2)(1 + 2) = 9$
$\sigma_n = 3(2r + 1)$
$\sigma_n = 3(2s + 1)$

$\sigma_n = (2r + 1)(2s + 1) = p*q = n$

Este último valor de $\sigma_n = n$ seguramente le llama la atención. Si n es el cuerpo de cifra y hay n mensajes que van en claro, quiere decir que ningún número se cifra, no sirve de nada nuestro sistema de cifra.

Al igual que con las claves privadas parejas, se consigue la menor cantidad posible de números no cifrables usando primos seguros. La cuestión ahora es ¿cuál será la cantidad mayor de estos NNC?

El peor de los escenarios será cuando:

$mcd (e - 1, p - 1) = p - 1$

$mcd (e - 1, q - 1) = q – 1$

En estas condiciones:

$\sigma_n = [1 + mcd (e - 1, p - 1)] [1 + mcd (e - 1, q - 1)] = [1 + (p -1)] [1 + (q - 1)]$

$\sigma_n = p * q = n.$

Así, todos los elementos del cuerpo n = p*q irán en claro. Esto nos indica que una vez elegidos p y q, hay que tener especial cuidado en la elección de la clave pública e. Otra razón para que el valor de la clave pública e no sea un número aleatorio elegido por el usuario generador de la clave sino un valor estándar como ya se ha visto.

Un ejemplo: suponga que tiene una clave RSA con p = 13, q = 17, $\phi(n) = 192$, n = 221 y e = 11. Si la genera, obtiene una clave óptima con 9 números no cifrables en el cuerpo n = 221. Pero si se hubiese elegido como clave pública el valor e = 49, igual de válido que 11, se observaría lo siguiente:

$mcd (e - 1, p - 1) = mcd (48, 12) = 12$

$mcd (e - 1, q - 1) = mcd (48, 16) = 16$

Por lo tanto $\sigma_{221} = [1 + 12][1 + 16] = 13*17 = 221 = n.$

¿Por qué ha pasado esto? En el ejemplo anterior, observamos que $e = 49 = 192/4 +1 = \phi(n)/4 + 1$. En general, si se elige un valor de clave pública válida $e = \phi(n)/k + 1$ para valores de k igual a 2, 3, 4, ... siempre pequeños, la cantidad de NNC puede ser muy alto, incluso todo el cuerpo de cifra como acaba de suceder para e = 49.

Si $\sigma(n) = n$, como todo el cuerpo de cifra irá en claro, entonces el valor de la clave pública se repetirá en la clave privada o en una clave privada pareja. Si llega a cumplirse que $e = \phi(n)/2 + 1$, entonces $\sigma(n) = n$ y será no cifrable cualquier valor de n. Es más, en este caso observará que la clave privada tendrá el mismo valor que la clave pública.

¿Hay que preocuparse por estos números no cifrables?

En el apartado anterior se ha visto que hay que tener cuidado en la elección del valor de la clave pública e, puesto que una mala elección de ésta podría llevarnos a una clave RSA en la que todos los valores a cifrar irían en claro. Para ello había que prestar especial atención a que el valor de la

clave pública e no coincidiese nunca con $\phi(n)/k + 1$ para $k = 2$, es decir $e = \phi(n)/2 + 1$, y con menos exigencias para k igual a 3, 4, 5, ... en general para valores de k bajos. Por suerte en la práctica la clave pública e no la elige el usuario sino que se fuerza a que sea el número 4 de *Fermat*, es decir 65.537 de solamente 17 bits. En estas condiciones, puesto que n y $\phi(n)$ serán números de mínimo 1.024 bits, entonces en el supuesto caso de que se cumpliese la relación $e = \phi(n)/k + 1$ el valor de k sería tan grande que la cantidad de números no cifrables nunca llegará a estos extremos.

3.5. Firma digital mediante el algoritmo RSA

Si recuerda las expresiones utilizadas para conseguir confidencialidad con RSA, eran las siguientes:

$$M^e \bmod n \quad | \quad C^d \bmod n$$

Una de las propiedades interesantes de RSA es que permite garantizar la autenticidad de un mensaje utilizando el mismo algoritmo simplemente utilizando la clave privada para cifrar y la clave pública para descifrar, es decir, al revés que cuando se garantiza la confidencialidad con este algoritmo. La idea es sencilla. Si el emisor cifra utilizando la clave privada, todo el mundo podrá descifrar ese información con la clave pública que es conocida, sin embargo, sólo el usuario que posee la clave privada es capaz de generar ese mensaje cifrado, por tanto, incorpora o garantiza autenticidad. Por tanto, el algoritmo RSA facilita la firma digital de mensajes.

En la mayoría de los escenarios reales la capacidad de garantizar confidencialidad y autenticidad se combinan para establecer sistemas criptográficos más robustos. Se verá mediante un ejemplo de comunicación entre un emisor (Alicia) y un receptor (Bernardo):

1) Alicia cifra un mensaje con destino a Bernardo.

$$C = M^{eB} \bmod n_B$$

2) Alicia firma el cifrado C con su clave privada para proporcionar autenticidad y envía $C_{firmado}$ a Bernardo.

$$C_{firmado} = C^{dA} \bmod n_A$$

3) Bernardo aplica la clave pública de Alicia para quitar o comprobar la firma y a continuación descifra el resultado del criptograma con su clave privada. Si todo es correcto, se recuperará el mensaje original enviado por Alicia.

$$M = (C_{firmado}{}^{eA} \bmod n_A)^{dB} \bmod n_B$$

Aunque es posible firmar digitalmente todo un documento entero, cifrando toda la información con la clave privada del emisor, en la práctica se suele acelerar este proceso firmando (cifrando) exclusivamente un resumen de la información a proteger. Este resumen se realiza mediante un algoritmo de hash. Un algoritmo de hash recibe como entrada una cantidad de n bits y produce una salida fija de bits, típicamente 128, 256 o 512 bits. Ejemplos de algoritmos de hash son MD5, SHA-1, SHA-2, etc. Si desea profundizar en estos algoritmos es interesante consultar algunas las referencias a este respecto en la sección de bibliografía.

Por tanto, una firma digital se realizará sobre un número, resultado de aplicar una función hash a un mensaje, ahora sí un archivo ya de cualquier tamaño y tipo. Según esto los pasos a realizar en la firma digital con RSA a través de un hash serían:

1) Alicia obtiene la función hash (un número) del archivo M que desea firmar.

2) Ese número debe representar al archivo de manera única, como si fuese una huella dactilar de él.

3) Sobre ese número o hash h(M) Alicia realizará la firma usando su clave privada d_A.

4) Alicia calcula $h(M)^{dA} \bmod n_A = s$ y le envía a Bernardo la dupla mensaje y firma (M, s).

5) Bernardo recibe (M', s) pues no se sabe si recibe el mismo mensaje M o un mensaje distinto o falso M'.

6) Bernardo realiza la siguiente operación con la clave pública de Alicia: $s^{eA} \bmod n_A = [h(M)^{dA} \bmod n_A]^{eA} \bmod n_A = h(M)$.

7) Es decir, recupera el valor h(M) que usó Alicia para su firma en el momento de emisión.

8) Bernardo calcula la función hash h(M') del mensaje M' recibido (puede ser el verdadero mensaje M u otro distinto M').

9) Si el hash calculado en recepción coincide con el hash recuperado y firmado en emisión, se acepta la firma y el mensaje como correcto.

En último lugar es importante resolver una cuestión que a veces pasa desapercibida. ¿Qué es mejor, primero cifrar y luego firmar o primero firmar y luego cifrar?

Habitualmente las aplicaciones suelen implementar la segunda opción, primero firmar digitalmente el mensaje a enviar y a continuación cifrar bien el mensaje o bien el mensaje más la firma. La opción de cifrar el mensaje a enviar y a continuación calcular la firma digital puede ser conveniente para minimizar el efecto de una negación de servicio en un entorno hostil. Por ejemplo, si un atacante desea sobrecargar al destino forzándole a hacer descifrados de mensajes falsos. Con este esquema, el destino podrá verificar la autenticidad del mensaje ahorrándose el proceso de descifrado, sólo se descifra la información una vez verificada la firma. Es decir:

1) A envía C (mensaje M cifrado con la clave pública de B) más firma digital de C (cifrado con clave privada de A el hash de C).

2) B descifra con la clave pública de A la firma digital y obtiene hash(C). Calcula hash(C'), siendo C' el mensaje cifrado recibido, y compara con hash (C). Si los hashes son iguales el mensaje es auténtico y procede a descifrarlo usando la clave privada de B. Si no son iguales descarta el mensaje evitándose descifrar la información para validar la autenticidad.

Firmar el mensaje cifrado tiene la ventaja de que se puede verificar la firma sin necesidad de descifrar el mensaje; esto podría minimizar ataques de negación de servicio provocados por mensajes generados con maldad para sobrecargar al destino al tener que descifrar cada mensaje para poder verificar la firma.

3.6. RSA y el teorema chino del resto

Los sistemas de cifra con clave pública, en general, son más lentos que los algoritmos de cifra simétrica. Esta característica se debe al tipo de operaciones que se deben realizar. Por ejemplo, los algoritmos asimétricos realizan operaciones del tipo elevar un número a una potencia (algoritmo RSA), mientras que los algoritmos simétricos (TDES, AES, etc.) típicamente realizan operaciones lógicas y desplazamientos de bits en registros.

Como la operación que se realiza para la cifra RSA es N^{clave} mod n, siendo n un cuerpo de al mínimo 1.000 bits producto de dos primos, y recomendable en la actualidad 2.048 bits, el talón de Aquiles estará en los valores que puedan tomar el número N que se cifra y la clave que se use en el exponente en dicha cifra dentro del cuerpo n.

Por ejemplo, si se cifra una clave de sesión de 256 bits (del algoritmo AES por ejemplo) con la clave pública de un servidor que tiene un módulo n de 2.048 bits, se obtendría un criptograma de 2.048 bits.

$$C = K_S^{eSv} \bmod n_{Sv} = 256 \text{ bits}^{17 \text{ bits}} \bmod 2.048 \text{ bits}$$

Dado que la clave privada del servidor d_{Sv} será un número muy cercano al cuerpo de cifra (2.048 bits) al elegir un e pequeño ($e = F_4$), la operación de descifrado será muy costosa en términos de cómputo. Recuérdese que la operación de cifrado es rápida al ser e pequeño (pocos bits a 1).

$$Ks = C^{dSv} \bmod n_{Sv} = 2.048 \text{ bits}^{2.048 \text{ bits}} \bmod 2.048 \text{ bits}$$

Es aquí donde entra en acción el teorema chino del resto, en tanto las dimensiones de todos los números que intervienen en la ecuación son muy grandes. El servidor, en vez de realizar el descifrado en el cuerpo n de 2.048 bits, al ser dueño de la clave y poseer los primos p y q, podrá realizar un conjunto de operaciones para el descifrado de ese criptograma pero en los cuerpos p y q de la mitad de tamaño en bits que el cuerpo de cifra n y, por tanto, reduciendo de una manera significativa el tiempo de cómputo.

Así, en nuestro ejemplo el servidor Sv realizará operaciones modulares en 1.024 bits, en vez de 2.048 bits, una diferencia inmensa. Se trata de un atajo que sólo posee el dueño de la clave y que optimizará de forma notoria el tiempo de cálculo, en teoría unas 4 veces más rápido. Software estándar como puede ser OpenSSL proporciona una serie de valores extras en la generación de claves RSA que precisamente son útiles para usar el TRC en el descifrado.

3.6.1. Cómo aplicar el TRC para ganar en eficiencia en aritmética modular

Si bien pueden encontrarse infinidad de sitios Web y documentos que traten este teorema, por ejemplo simplemente consultando en Google, lo cierto es que no existen demasiados documentos que expliquen de forma detallada y clara porqué puede usarse el TRC en una expresión del tipo A^B mod n, cuando n es el producto de dos primos p y q como es el caso de RSA.

A continuación se presenta una detallada explicación de este asunto, partiendo del trabajo de la profesora Dña. *Ana Isabel Lías Quintero*, del Departamento de Matemática Aplicada de la EUI - UPM. Las notaciones serán algo diferentes a las utilizadas pero sin embargo son muy fáciles de seguir.

La complejidad computacional del cálculo del texto en claro $M = C^d \bmod n$ depende de d y n. De hecho, es O (lg d * lg^2 n). Los tamaños de n y d son los que determinan la complejidad. Si k = max {tam (n), tam (d)}, la complejidad es O (k^3).

En lo siguiente se verá una forma de reducir el número de operaciones del cálculo de $C^d \bmod n$; para ello será fundamental el teorema chino del resto. En el caso de RSA, n = p * q, siendo p y q primos, y el enunciado del teorema quedaría:

Sean p, q primos distintos y n = p * q. Sean $x_1, x_2 \in \mathbb{Z}$.
El sistema:

$$x \equiv x_1 \bmod p$$
$$x \equiv x_2 \bmod q$$

Tiene solución, que además es única módulo n. Dicha solución es de la forma:

$$x = [x_1 * q * (q^{-1} \bmod q) + x_2 * p * (p^{-1} \bmod p)] \bmod n$$
siendo: $q^{-1} = inv (q, p); p^{-1} = inv (p, q)$

Hay otro resultado que, combinando con el TRC, es de gran utilidad.

Propiedad 1: $a \equiv b \bmod n \Rightarrow a \equiv b \bmod d$ para todo *d* divisor de *n*

Por ejemplo, si n = 585 = 3^2 * 5 * 13, entonces dado que el resto de dividir 1.263 veces por 585 es 93, puede concluirse que: 1.263 ≡ 93 mod 585. Si se toma ahora el número 65 = 5 * 13, divisor de 585: 1.263 ≡ 28 mod 65 y 93≡ 28 mod 65. Por tanto, 1263 y 93 son congruentes mod 65, por transitividad.

Conjugando ambas propiedades, se obtiene el siguiente corolario:

Propiedad 2: Sea n = p * q con p y q primos distintos

Entonces, para todo $a, b \in \mathbb{Z}$

$$a \equiv b \bmod n \Longleftrightarrow a \equiv b \bmod p \text{ y } a \equiv b \bmod q$$

La aplicación de la propiedad 2 es una primera simplificación del cálculo de $M = C^d \bmod n$. Se resuelve esta ecuación realizando las siguientes operaciones:

$$M \equiv C^d \bmod p \text{ (I)}$$
$$M \equiv C^d \bmod q \text{ (II)}$$

Ventajas:

1) En lugar de hacer cálculos mod n, éstos se hacen módulo p y q, cuyos tamaños son del orden de la mitad del de n.

2) Los cálculos de (I) y (II) se pueden hacer en paralelo.

¿Hay más reducciones? Sí, hay dos más. Las propiedades matemáticas en las que se basan son las siguientes.

Propiedad 3: Sea p primo y mcd (a, n) = 1. Entonces, si $r \equiv t \bmod \phi(p) \Rightarrow a^r \equiv a^t \bmod p$. Recuerda que como p es primo, $\phi(p) = p - 1$.

La aplicación de esta propiedad 3 nos permite una segunda simplificación del cálculo de $M = C^d \bmod n$:

$C^d \bmod p \equiv C^{d \bmod (p-1)} \bmod p \equiv Cp^{d \bmod (p-1)} \bmod p$ (con Cp = C mod p)
$C^d \bmod q \equiv C^{d \bmod (q-1)} \bmod q \equiv Cq^{d \bmod (q-1)} \bmod q$ (con Cq = C mod q)

Si dp = d mod (p-1) y dq = d mod (q-1), entonces:
$C^d \bmod p \equiv Cp^{dp} \bmod p$
$C^d \bmod q \equiv Cq^{dq} \bmod q$

Ventajas:

1) Aunque el tamaño de la clave privada d es del orden de tamaño del módulo n, los tamaños de dp y dq son del orden de la mitad.

2) Además, dichos valores pueden calcularse una sola vez y tenerlos almacenados para descifrados posteriores.

Por último, para calcular el inverso modular puede usarse el Pequeño Teorema de *Fermat* que se indica como propiedad 4.

Propiedad 4: Sea p primo y mcd (a, p) = 1. Entonces $a^{-1} = \text{inv}(a, p) \equiv a^{p-2} \bmod p$

Retomando la ecuación del TRC:

$$x = [x_1 * q * (q^{-1} \bmod q) + x_2 * p * (p^{-1} \bmod p)] \bmod n$$

en donde:

$x_1 = Cp^{dp} \bmod p$
$x_2 = Cq^{dq} \bmod q$

$p^{-1} = p^{p-2} \bmod p$
$q^{-1} = q^{q-2} \bmod q$

Un último paso: la fórmula de Garner

Existe una última optimización basada en la siguiente regla conocida como fórmula de *Garner* propuesta por el matemático Harvey L. *Garner*.

Para obtener x mod n con n = p * q, a partir de $x \equiv x_1$ mod p y $x \equiv x_2$ mod q, hay que hacer:

$$x = x_1 + h * p, \text{ en donde } h = [(x_2 - x_1)(p^{-1} \bmod q)] \bmod q$$

Conocidas todas estas propiedades, puede documentarse el procedimiento final y la complejidad asociada para calcular $M = C^d$ mod n, con n = p * q.

1. Precalcular estos valores:
   1. dp = d mod (p-1)
   2. dq = d mod (q-1)
   3. p^{-1} mod q = p^{q-2} mod q

2. Realizar operaciones de descifrado con el criptograma C:

   Paso 0: Calcular Cp = C mod p; Cq = C mod q

   Paso 1: Calcular $x_1 = Cp^{dp}$ mod p

   Paso 2: Calcular $x_2 = Cq^{dq}$ mod q

   Paso 3: Calcular $h = [(x_2 - x_1)(p^{-1} \bmod q)] \bmod q$

   Paso 4: Devolver $x = x_1 + h * p$

En cuanto a la complejidad, decir lo siguiente:

1. Los pasos más costosos son el 1 y el 2.
2. El paso 1 tiene una complejidad del orden $(k/2)^3$
3. El paso 2 tiene una complejidad del orden $(k/2)^3$
4. Los pasos 0, 3 y 4 son sumas y productos, que tienen una complejidad a lo sumo cuadrática en k.

Por lo tanto, sin contar con el preprocesamiento:

1. Si los pasos 1 y 2 se paralelizan, este método es del orden de 8 veces más rápido.
2. Si los pasos 1 y 2 no se paralelizan, este método es del orden de 4 veces más rápido. Esto es: $2(k/2)^3 = 2(k/8) = k/4$

Para un estudio más formal de éste y otros temas matemáticos relacionados con RSA, se recomienda el documento Fundamentos matemáticos del método RSA del profesor *Hugo Scolnik* de la Universidad de Buenos Aires.

3.6.2. Aplicación del TRC en el descifrado de RSA

Para el sistema RSA pueden utilizarse dos representaciones del teorema chino del resto, la primera como fórmula estándar y que se presenta a continuación o la que se deriva de aplicar la fórmula de *Garner* ya vista, puesto que ambas son lo mismo.

TRC con fórmula estándar

$$M = \{A_p[C_p^{dp} \text{ (mod p)}] + A_q[C_q^{dq} \text{ (mod q)}]\} \text{ mod n}$$

siendo:

$A_p = q\ [\text{inv}\ (q, p)] = q^{p-1} \text{ mod n}$

$A_q = p\ [\text{inv}\ (p, q)] = p^{q-1} \text{ mod n}$

Observa que estas dos reducciones no se habían visto en el apartado anterior.

$dp = d \text{ mod } (p-1);\ dq = d \text{ mod } (q-1)$

$C_p = C \text{ mod } p;\ C_q = C \text{ mod } q$

Un ejemplo de uso. Se recibe como criptograma el valor decimal 582.819 y se pide descifrarlo usando el TRC, sabiendo que los datos privados del dueño de la clave son: p = 859, q = 907 (n = 779.113) y d = 17.249.

Solución:

$$M = \{A_p[C_p^{dp} \text{ (mod p)}] + A_q[C_q^{dq} \text{ (mod q)}]\} \text{ mod n}$$

$A_p = q^{p-1} \text{ mod n} = 907^{859-1} \text{ mod } 779.113 = 470.733$ (precalculado)

$A_q = p^{q-1} \text{ mod n} = 859^{907-1} \text{ mod } 779.113 = 308.381$ (precalculado)

$dp = d \text{ mod } (p-1) = 17.249 \text{ mod } (859 - 1) = 89$ (precalculado)

$dq = d \text{ mod } (q-1) = 17.249 \text{ mod } (907 - 1) = 35$ (precalculado)

$C_p = C \text{ mod } p = 582.819 \text{ mod } 859 = 417$

$C_q = C \text{ mod } q = 582.819 \text{ mod } 907 = 525$

Reemplazando:

$M = [470.733 * 417^{89} \text{ mod } 859] + [30.8381 * 525^{35} \text{ mod } 907] \text{ mod } 779.113$

$M = [470.733 * 322 + 308.381 * 824] \text{ mod } 779.113$

$M = [151.576.026 + 254.105.944] \text{ mod } 779.113$

$M = 405.681.970 \text{ mod } 779.113 = 543210$, que es el mensaje que se observa al descifrar el criptograma 582.819.

■■■

TRC optimizado con fórmula de *Garner* ($x_1 = m_1$ y $x_2 = m_2$)

$m = m_2 + h * q$

$h = qInv * [(m_1 - m_2)] \bmod p$

siendo:

$m_1 = C^{dp} \bmod p$

$m_2 = C^{dq} \bmod q$

dp = d mod (p-1) (precalculado)

dq = d mod (q-1) (precalculado)

qInv = q^{-1} mod p (precalculado)

Usando ahora la fórmula de *Garner* para descifrar el criptograma 3.108.635 en un sistema RSA en el que p = 3.449, q = 3.061 (n = 10.557.389), d = 2.335.691:

$m = m_2 + h * q$

$h = qInv * [(m_1 - m_2)] \bmod p$

dp = d mod (p-1) = 2.335.691 mod (3.449 - 1) = 1.395 (precalculado)

dq = d mod (q-1) = 2.335.691 mod (3.061 - 1) = 911 (precalculado)

qInv = q^{-1} mod p = inv (3.061, 3.449) = 80 (precalculado)

$m_1 = C^{dp} \bmod p = 3.108.635^{1.395} \bmod 3.449 = 1.342$

$m_2 = C^{dq} \bmod q = 3.108.635^{911} \bmod 3.061 = 993$

Reemplazando:

h = 80 * [1.342 - 993] mod 3.449 = 27.920 mod 3.449 = 328

m = 993 + 328 * 3.061 = 1.005.001

que es el mensaje que se obtiene al descifrar el criptograma 3.108.635. Compruebe ambos resultados con el software Fortaleza de cifrados que encontrará en el "Apéndice C. Software educativo".

Con estos ejemplos queda clara la utilidad del uso del teorema de restos chinos en RSA. No obstante, este teorema también es de utilidad en muchos otros protocolos y algoritmos criptográficos. Por ejemplo, el protocolo conocido como transferencia inconsciente o trascordada presentado por

Michael Rabin en 1979, también conocido como Criptosistema de *Rabin*, y que forma parte medular de un buen número de protocolos criptográficos como el problema del lanzamiento de la moneda, el problema de la firma de contratos, el problema de los prisioneros, etc. Mención especial requiere el algoritmo esquema de umbral de *Shamir* para compartir secretos usando el teorema chino del resto, aunque sea más conocida la solución que utiliza la interpolación de *Lagrange*. Para una aproximación a este esquema puede verse el vídeo "Protocolo de reparto de secretos" de la enciclopedia visual de la seguridad de la información Intypedia (véase la sección de bibliografía).

3.6.3. Precauciones en el uso del TRC en RSA

Como se ha visto en apartados anteriores es posible acelerar los cálculos en RSA mediante el Teorema del Resto Chino, lo que se traduce en pasar de calcular una potencia modular en un módulo enorme (módulo n) a dos operaciones modulares de tamaño la mitad del primero en bits (tamaño de p y tamaño de q). Esto hoy día es muy útil especialmente en aplicaciones con recursos limitados como pueden ser dispositivos de bajo coste, dispositivos embebidos, etc. Un ejemplo muy socorrido son las tarjetas inteligentes (smartcards).

Aunque RSA es un algoritmo seguro, se profundizará en esta cuestión en el capítulo destinado a su seguridad, no se debe obviar toda una serie de ataques posibles cuando se tiene acceso físico al dispositivo que realiza los cálculos matemáticos en la generación de claves o en el cálculo del cifrado/descifrado. Un ejemplo en ataques en escenarios como pueden ser las smartcards o similares son por ejemplo los ataques de *power glitching* (variaciones en la tensión de alimentación) que pueden hacer que el procesador malinterprete instrucciones o datos, o incluso se las salte. Por ejemplo, si se baja repentinamente la tensión puede darse el caso que un bit con valor '1' llegue con un valor por debajo del umbral de decisión, de forma que se lea como un '0'. Así se consigue producir un valor incorrecto o conseguir que se ejecute un salto condicional cuando no debería. Por otra parte, existen otros métodos como variar la temperatura del chip de forma que quede fuera de los umbrales de funcionamiento definidos por el fabricante, mediante radiación electromagnética o utilizando fuentes de luz ya sea láser, rayos X o incluso luz blanca, aunque estos últimos ataques son invasivos pues hay que abrir el chip para llegar a introducir la luz en el punto exacto[3].

Para comprender mejor estos ataques se va a analizar un ejemplo de ataque basado en inyección de fallos, *Bellcore attack*, en el que se demuestra que es factible recuperar la clave privada, utilizada por ejemplo para firmar una información.

Supongamos que utilizando RSA-CRT deseamos firman un mensaje m, produciendo la firma S (el resultado cifrado), adaptando las fórmulas indicadas anteriormente:

$$S_p = m_p^{dp} \bmod p$$
$$S_q = m_q^{dq} \bmod q$$
$$S = ((S_q - S_p) * p_{inv}) \bmod q) * p + S_p$$
$$\text{donde } m_p = m \bmod p,\ m_q = m \bmod q,\ p_{inv} = p^{-1} \bmod q$$

3 Este resumen está extraído del blog del investigador español Eloi Sanfelix especialista en este campo, blog de recomendada lectura por su habilidad en tratar de forma amena estos temas. Se puede ver sus artículos en http://www.limited-entropy.com/

Supongamos que es viable inyectar un fallo en S_p o S_q y que el atacante puede conocer el resultado de la firma sin fallo (S) y el resultado al inyectar el fallo (S'). Si esto es así es posible recuperar la clave privada calculando:

$$mcd(S-S',N)$$

o recuperarla desde S', exponente e, N y m:

$$mcd(m-(S'^e \bmod N), N)$$

ya que el primo p o q será el único factor común. Conocido un primo y N puede deducirse el otro primo y por tanto adivinar la clave.

Así pues, mediante un poco de matemáticas y una inyección de un fallo en un resultado resulta muy sencillo obtener la clave RSA. Esto deja clara la necesidad de proteger los dispositivos embebidos ante este tipo de ataques. Si desea profundizar en esta temática puede consultar referencias en la sección de bibliografía.

3.7. Software OpenSSL. Practicando

Basado en la librería *SSLeay* desarrollada por *Eric A. Young* y *Tim Hudson*, OpenSSL es un proyecto open-source escrito en lenguaje C que implementa un buen número de funciones criptográficas, entre ellas las que dan soporte al algoritmo RSA, como puede ser la generación de claves. Una característica interesante de este software es su condición multiplataforma y su constante actualización.

En lo que sigue se va a describir cómo utilizar RSA con software criptográfico real, como es OpenSSL. Los ejemplos adjuntados se han realizado mediante la versión para sistema operativo Windows de este proyecto. Si desea probar este software en otras plataformas o software parecido es posible que muchas de las cosas indicadas en este apartado pudieran ser de utilidad.

Este apartado aunque basado en software real tiene por objetivo reforzar conceptos estudiados y no ser un mero manual de uso de una herramienta que con el tiempo, como todo, quedaría obsoleto. Si desea utilizar openSSL para cifrar/descifrar/firmar información en Internet encontrará buenos manuales en Internet a tal fin.

3.7.1. Generación de claves RSA con OpenSSL

El comando para generar claves RSA de OpenSSL que aparece en su Web es el siguiente:

openssl genrsa [-out filename] [-passout arg] [-des] [-des3] [-idea] [-f4] [-3] [-rand file(s)] [-engine id] [numbits]

Como primer ejercicio, generará una clave de 1.024 bits de nombre c4yRSA1 como se observa en la siguiente figura. Para ello debe usar el siguiente comando:

```
C:\OpenSSL-Win32\bin>openssl genrsa -out c4yRSA1 1024
```

```
DOS

C:\OpenSSL-Win32\bin>openssl genrsa -out c4yRSA1 1024
Loading 'screen' into random state - done
Generating RSA private key, 1024 bit long modulus
.....++++++
..........++++++
unable to write 'random state'
e is 65537 (0x10001)

C:\OpenSSL-Win32\bin>
```

Figura 18. Generación de la clave c4yRSA1 de 1024 bits.

Seguramente le llamará la atención esos puntos y esas cruces que aparecen en dos líneas mientras se calcula la clave. Se trata de la indicación que nos hace el programa que está intentando encontrar un primo p y luego un primo q, y comprobando que ambos son primos. Más adelante tendrá ocasión de comprobarlo cuando calcule una clave muy grande y por tanto tarde bastante tiempo en generarse. Además puede comprobar que si no se especifica nada al respecto, OpenSSL usa como clave pública estándar el número 4 de *Fermat*: 65.537 ó 0x10001.

Ejecutando el siguiente comando comprobará que para generar esta clave de 8.192 bits el programa tarda varios segundos, lo que se observa en la velocidad en que van apareciendo los puntos en la pantalla.

C:\OpenSSL-Win32\bin>openssl genrsa -out muygrande 8192

```
Administrador: DOS

C:\OpenSSL-Win32\bin>openssl genrsa -out muygrande 8192
Loading 'screen' into random state - done
Generating RSA private key, 8192 bit long modulus
.............................................................................
.............................................................................
.............................................................................
...............................++
..........................................................................++
e is 65537 (0x10001)

C:\OpenSSL-Win32\bin>_
```

Figura 19. Generación de la clave muygrande de 8192 bits.

Si ha estado atento a los dos ejercicios anteriores, se habrá dado cuenta que en el primero el programa nos entregaba un mensaje de error diciendo *unable to write 'random state'* y en el segundo ese mensaje ya no aparecía. Es más, seguramente habrá obtenido en ambas claves que ha generado el mismo mensaje de error.

Esto se debe a que para la primera clave se ha ejecutado OpenSSL en Windows sin ser administrador del sistema y en el segundo ejemplo sí se ha hecho como administrador. El mensaje de error nos indica que OpenSSL no puede abrir un archivo de semilla, aunque esto no sea un problema para que se genere la clave. Puede leer varias respuestas sobre esto en Internet o bien desde las FAQs de OpenSSL.

Para no recibir ese mensaje de error, debe entrar como administrador o, más cómodo, ejecutar DOS como administrador utilizando para ello el botón derecho del ratón en el icono de sistema como se muestra en la siguiente figura.

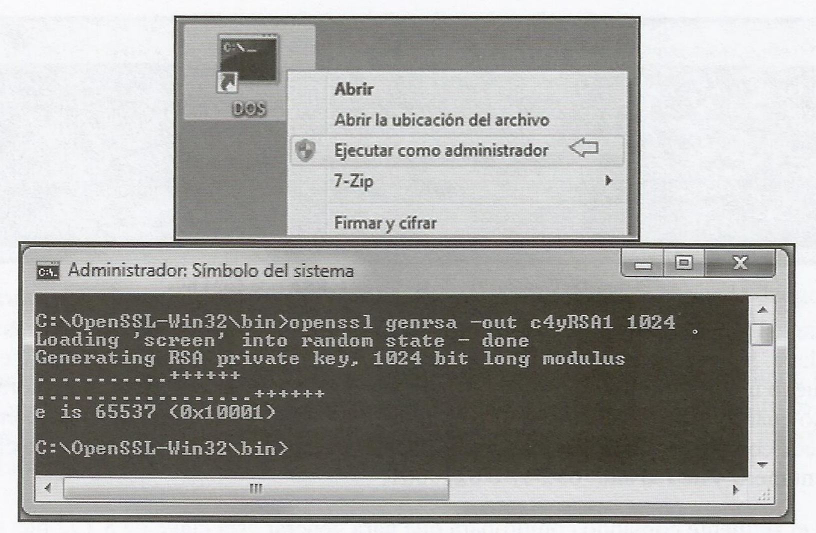


Figura 20. Generación de la clave c4yRSA1 de 1024 bits como administrador.

Generada una clave, podrá abrirla por ejemplo con WordPad para ver su valor en representación Base64. Si no ejecuta ningún otro comando en OpenSSL, sólo mostrará la clave privada como se observa en la figura siguiente.

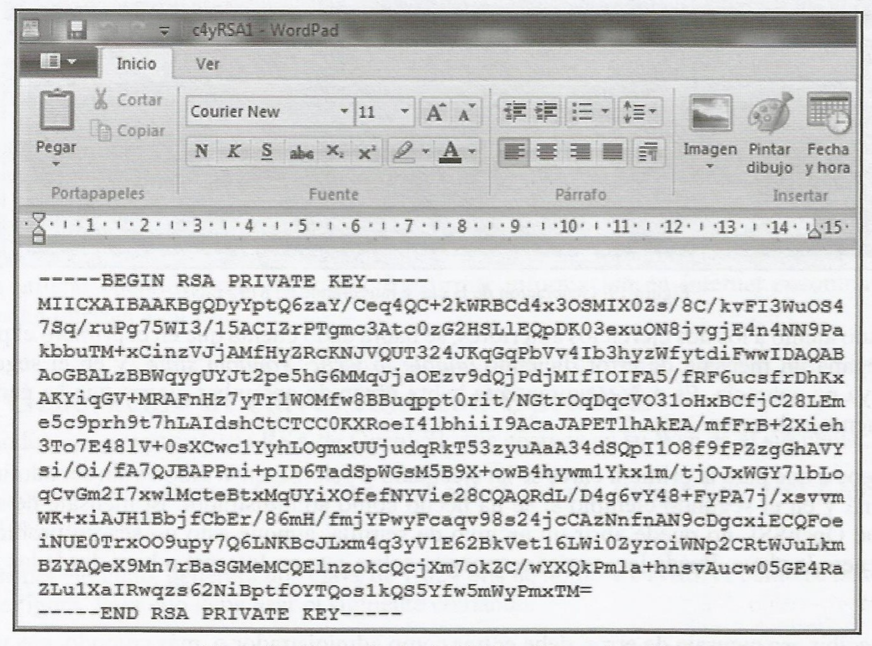


Figura 21. Clave privada c4yRSA1 en formato base64.

Para ver la clave pública, debe ejecutar antes el siguiente comando:

C:\OpenSSL-Win32\bin>openssl rsa -in c4yRSA1 -pubout

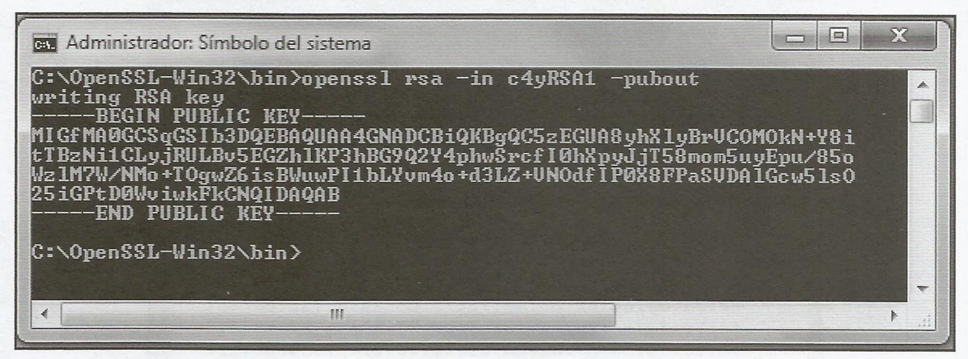


Figura 22. Exportación de la clave pública c4yRSA1 en formato base64.

Conversión de claves RSA binarias a formato texto

Convertir la clave pública RSA a formato texto Base64 tiene sentido porque nos permite exportarla y darla a conocer a todo el mundo. La pregunta que se puede estar haciendo en este momento es qué interés existe en convertir la clave RSA de formato binario a formato texto. La respuesta es simple. Con esta transformación se obtiene un archivo en donde se mostrarán todos los parámetros de esa clave en formato hexadecimal; es decir, los números p, q, n, e y d. Además comprobará que hay otros tres parámetros que permitirán usar en el descifrado el teorema chino de los restos.

Puede usar entonces esos valores de los primos p, q y la clave pública e entregada por OpenSSL como entrada al software genRSA y generar manualmente esa clave. Entre otras cosas, esto permitirá comprobar si OpenSSL usa o no primos seguros en la generación de tales claves, especialmente cuando genere de forma automática varias claves.

Generada la clave c4yRSA1 de 1024 bits con OpenSSL, para convertirla a formato texto (hexadecimal) se usa el siguiente comando:

C:\OpenSSL-Win32\bin>openssl rsa -in c4yRSA1 -text -out c4yRSA1Texto

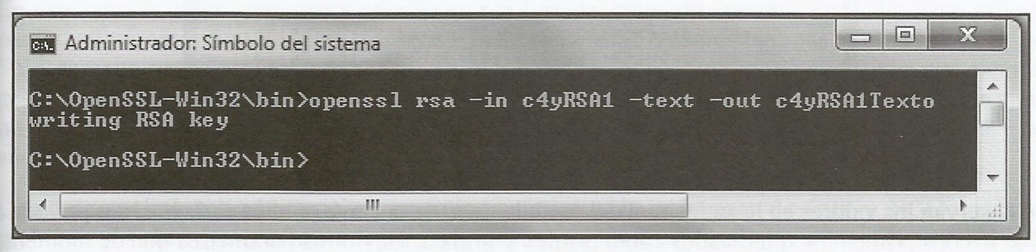


Figura 23. Conversión de la clave c4yRSA1 a formato hexadecimal.

Al abrir el archivo c4yRSA1Texto por ejemplo con WordPad (para mantener el formato) se obtiene un documento como éste:

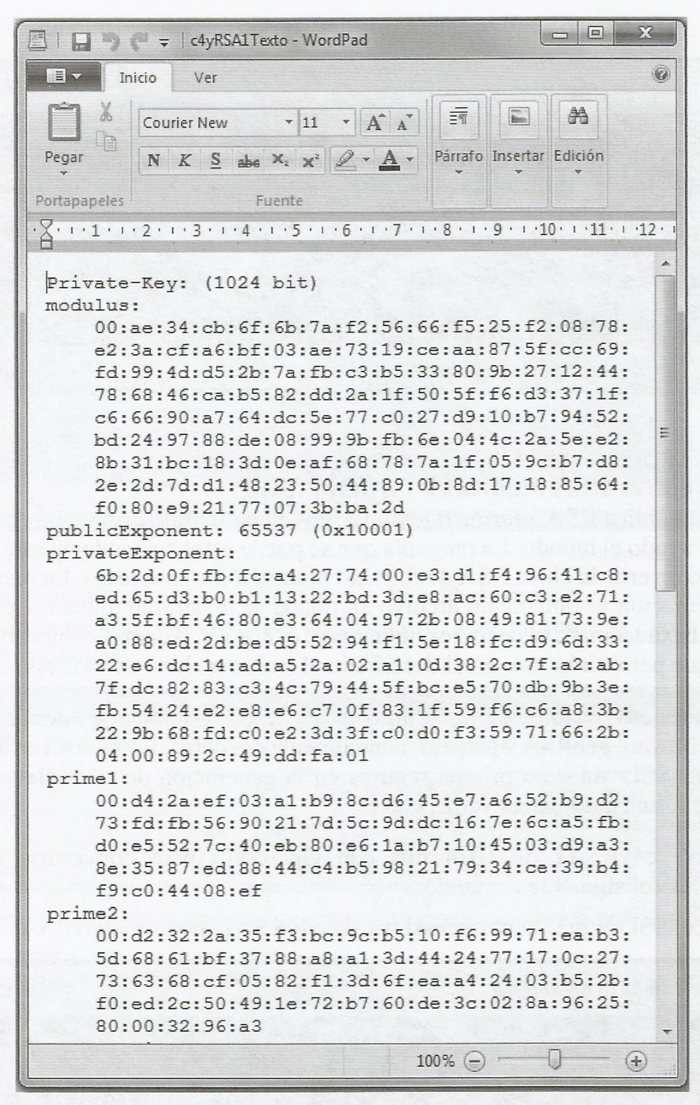


Figura 24. Clave c4yRSA1Texto abierta con Wordpad.

Se observan los valores en hexadecimal del módulo de n, de la clave pública e, de la clave privada d y de los dos primos p y q indicados como prime1 y prime2. Hay otros parámetros que de momento no se analizarán.

Para poder utilizar estos valores es necesario eliminar los ":" que separan los bytes así como la sangría de cuatro espacios que se observa a la izquierda. Estos parámetros (p, q y e) se podrían introducir en un software de generación de claves, como el programa genRSA, y obtener la clave propuesta por OpenSSL.

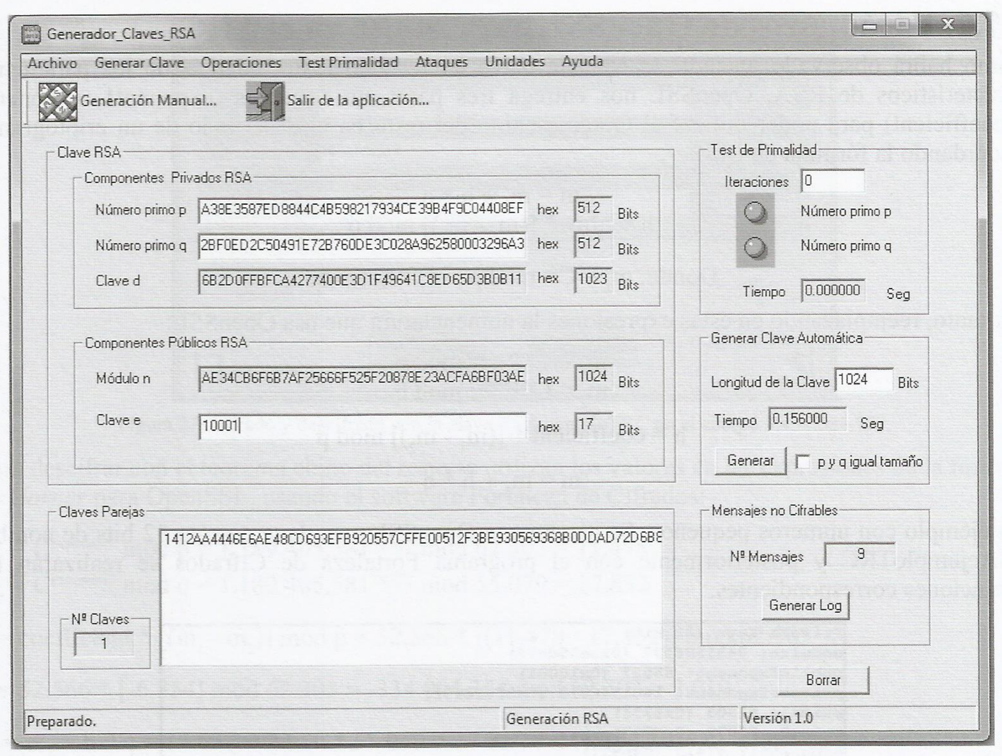


Figura 25. c4yRSA1 generada manualmente por genRSA con los valores aportados por OpenSSL.

Como puede apreciar, indicando en el comando solamente que el tamaño de la clave era 1.024 bits, OpenSSL ha generado una clave RSA con dos primos p y q cada uno de 512 bits, un módulo n de 1.024 bits, la clave pública e estándar con el número 4 de *Fermat* 0x10001 y una clave privada d con un tamaño de 1023 bits.

En este caso la clave generada es de las que se han denominado óptimas, con una clave privada pareja y 9 números no cifrables. ¿Generará siempre OpenSSL claves óptimas? La respuesta es no. Sólo necesita jugar un poco con OpenSSL y el software educativo genRSA. Pruebe por ejemplo generar una clave de 1.024 bits con e = 0x10001 con OpenSSL; tarde o temprano obtendrá una clave no óptima, con CPP > 1 o bien NCC > 9. Tenga en cuenta que genRSA usa la librería *Crypto++* del año 2004; las versiones actuales de OpenSSL ya no generan tantas claves no óptimas.

Como se analizó sobradamente en capítulos anteriores, la existencia de CPP y NCC no es un problema si se tiene en cuenta las diversas recomendaciones reflejadas, entre ellas utilizar un tamaño de n como mínimo de 1.024 bits. A modo de curiosidad, ¿cuál es el tamaño de clave más grande que permite crear OpenSSL? ¿Sería posible crear una clave de 64.000 bits? Como ejercicio, intente crear una clave de 24.000 bits con OpenSSL y observe lo que tarda el programa en generarla. Deberá eso sí tener mucha paciencia y tiempo porque podría tardar varias horas. Si tiene curiosidad sobre este tema de la longitud máxima de una clave RSA en la sección de bibliografía puede consultar referencias de interés.

■ ■ ■

3.7.2. Parámetros de OpenSSL para su uso en el TRC

Como habrá observado, cuando se edita la clave en formato texto, además de los parámetros característicos de RSA, OpenSSL nos entrega tres parámetros nuevos (exponent1, exponent2 y coefficient) para poder utilizar el teorema chino del resto en el descifrado de un criptograma. Recordando la fórmula de *Garner*:

$$m = m_2 + h * q$$
$$h = qInv * [(m_1 - m_2)] \bmod p$$

$$\text{Donde: } m_1 = C^{dp} \bmod p \; ; \; m_2 = C^{dq} \bmod q$$

Por tanto, reemplazando en estas expresiones la nomenclatura que usa OpenSSL:

$$m_1 = C^{exponent1} \bmod p$$
$$m_2 = C^{exponent2} \bmod q$$

$$h = coefficient * [(m_1 - m_2)] \bmod p$$

$$m = m_2 + h * q$$

Un ejemplo con números pequeños. Se genera con OpenSSL una clave de sólo 32 bits de nombre RSAejemploTRC y posteriormente con el programa Fortaleza de Cifrados se realizarán las operaciones correspondientes.

```
Private-Key: (32 bit)
modulus: 3431586937 (0xcc89dc79)
publicExponent: 65537 (0x10001)
privateExponent: 1601146513 (0x5f6f8e91)
prime1: 62303 (0xf35f)
prime2: 55079 (0xd727)
exponent1: 47415 (0xb937)
exponent2: 29053 (0x717d)
coefficient: 52566 (0xcd56)
-----BEGIN RSA PRIVATE KEY-----
MC0CAQACBQDMidx5AgMBAAECBF9vjpECAwDzXwIDANcnAgMAuTcCAnF9AgMAzVY=
-----END RSA PRIVATE KEY-----
```

Figura 26. Clave RSAejemploTRC en formato texto.

Observe que al ser muy pequeño el tamaño de la clave, OpenSSL nos la muestra en formato decimal y hexadecimal. Se cifra con esta clave el valor 111.222.333 como muestra la figura.

Exponenciación Modular
Base 111222333 9 digitos
Exponente 65537 5 digitos
Modulo 3431586937 10 digitos
Resto Modular 1189405581 10 digitos
Calcular Cancelar
0 h: 0 m: 0 s:

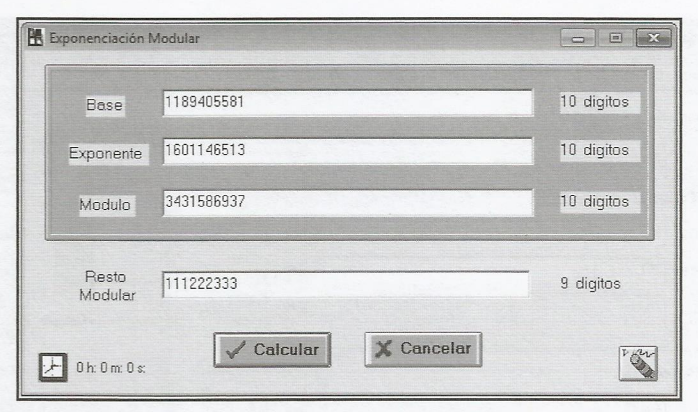


Figura 27. Cifrado y descifrado con la clave RSAejemploTRC en el cuerpo 3.431.586.937.

Para descifrar con el teorema chino del resto se utilizan los valores en las expresiones de la fórmula de *Garner* para OpenSSL, usando el software Fortaleza de Cifrados:

$m_1 = C^{exponent1} \bmod p = 1.189.405.581^{47.415} \bmod 62.303 = 11.478$
$m_2 = C^{exponent2} \bmod q = 1.189.405.581^{29.053} \bmod 55.079 = 17.832$

$h = coefficient * [(m_1 - m_2)] \bmod p = 52.566 * [(11.478 - 17.832)] \bmod 62.303$

$h = 52.566 * [-6.354] \bmod 62.303 = -334.004.364 \bmod 62.303 = 2.019$

$m = m_2 + h * q = 17.832 + 2.019 * 55.079 = 17.832 + 111.204.501 = 111.222.333$

3.8. Ejercicios y prácticas

Ejercicio 1. Generación de claves RSA

Genere una clave RSA donde p = 197, q = 251, e = 19.

1) Se ejecuta el programa genRSA y se introducen estos valores en las casillas de p, q y e, usando la opción copiar y pegar.

2) Se pegan los valores 197, 251 y 19.

3) Desde la parte superior izquierda de la aplicación, se pulsa Generación Manual.

4) Se obtiene la clave que aparece en la siguiente figura, donde: La clave pública es n = 49.447 y e = 19, siendo la clave privada d = 2.579.

5) Se observan los valores de las Claves Privadas Parejas y los Mensajes No Cifrables que muestra el programa genRSA.

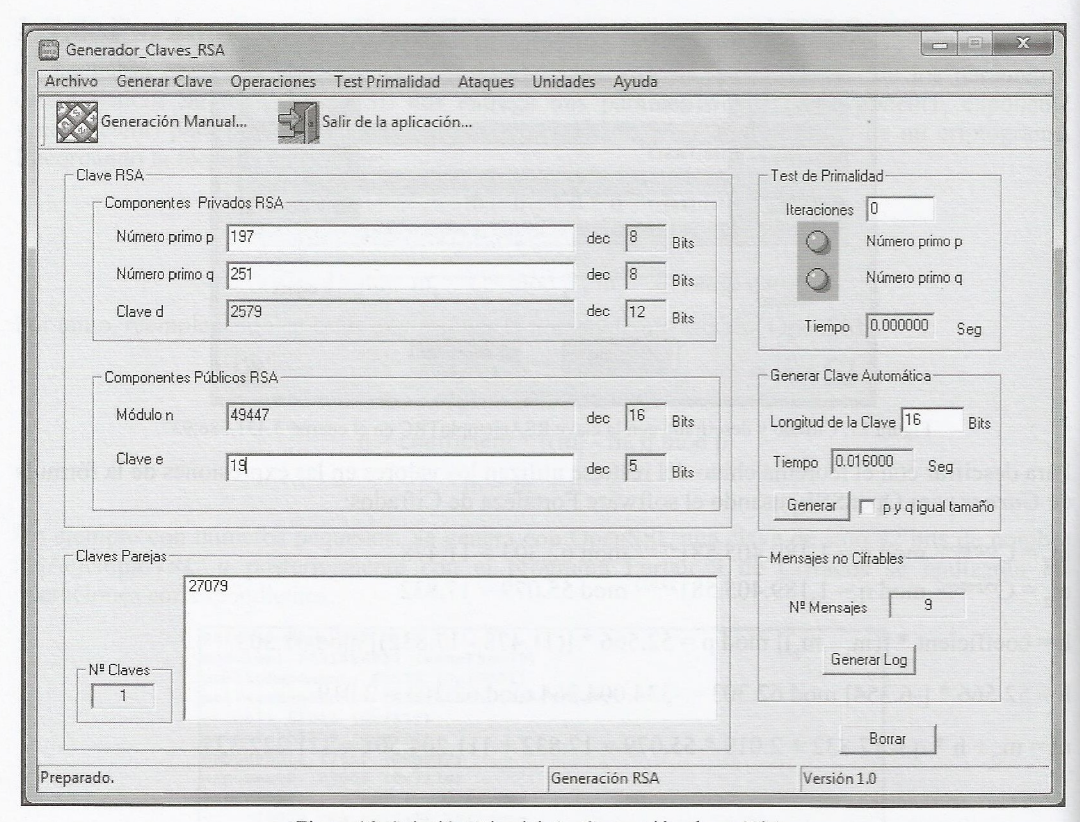


Figura 28. Solución: Ejercicio1 - Generación claves RSA.

Ejercicio 2. Operaciones de cifrado y descifrado RSA

Alicia tiene como clave pública RSA los valores nA = 5.963 y eA = 13. Bernardo desea enviarle de forma confidencial el número secreto N = 101. Indique las operaciones de cifrado y descifrado y usa para comprobarlo el software genRSA.

1) Bernardo hace la siguiente operación N_A^e mod n_A = 10113 mod 5.963 = 3.726.

2) Como n = 5.963, es fácil comprobar que pA = 67 y qA = 89.

3) Por tanto φ (nA) = 66 x 88 = 5.808 y d = inv (13, 5.808) = 4.021.

4) Puede realizar este cálculo usando la calculadora de su sistema operativo favorito.

5) Alicia recibe el criptograma C = 3.726 y realiza la operación CdA mod nA.

6) CdA mod nA = 3.7264.021 mod 5.963 = 101.

7) Puedes realizar también este cálculo usando la calculadora.

8) Alicia recupera el valor secreto 101 enviado por Bernardo.

La siguiente figura muestra las operaciones de cifrado y descifrado con el programa genRSA. Recuerda que este programa sólo realiza la cifra usando la clave pública.

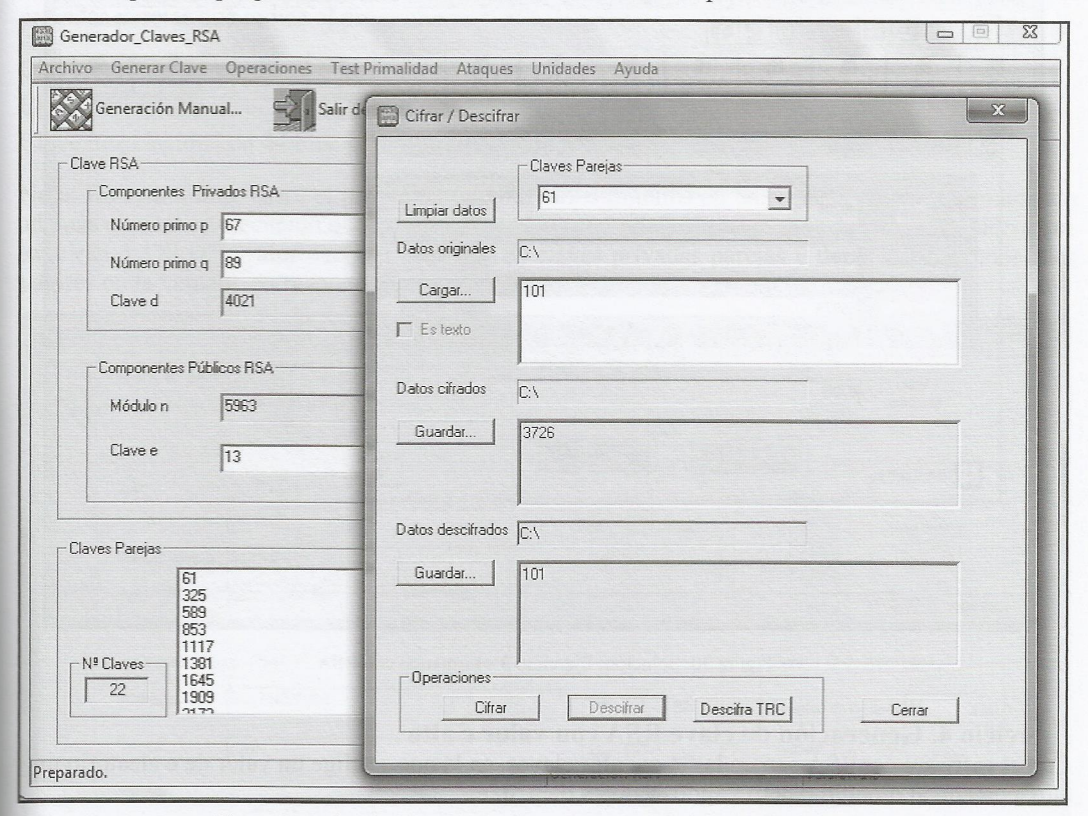


Figura 29. Solución: Ejercicio 2 - Operaciones de cifrado y descifrado RSA.

Ejercicio 3. Firmando con RSA

Bernardo desea enviarle a Alicia firmado el valor 40.205. La clave pública de Bernardo es nB = 55.973 y eB = 17, y su clave privada dB = 22.853. Puesto que Md = 40.20522.853 mod 55.973. Para resolverlo se usa el software Fortaleza de Cifrados.

1) Se ejecuta el programa Fortaleza y en la barra de iconos se pulsa en las herramientas, arriba a la izquierda.

2) Se elige la operación potencia.

3) Se introducen los valores 40205, 22853 y 55973.

4) Se obtiene como resultado de firma el valor 6306.

5) Con el mismo software, usando ahora la clave pública de Bernardo e = 17, se comprueba que el valor firmado es 40.205.

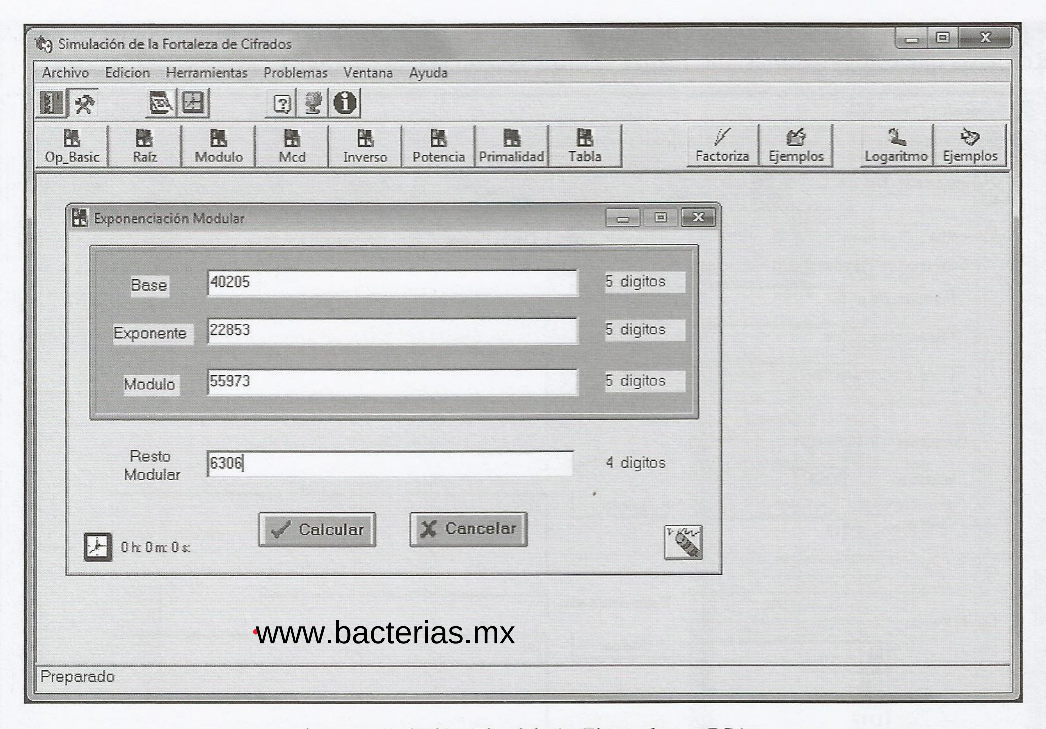


Figura 30. Solución: Ejercicio 3 - Firmando con RSA.

Ejercicio 4. Generación de clave RSA con valor e alto

Con el software genRSA genere las siguientes claves, en la que se elige un valor de e aleatorio muy alto, cercano al módulo n pero válido. Valores de p, q, e:

```
EE0F219CCB4BCDB17AB66B7D692F6E72D25CEE16B293FE06C3FB3A3BABEE9FCF
CBE21DB30DD956EDF1BD43A0AB58530FBF35B34465B614BA7761AE5A8E7B8637
1B15C0A95169D14B68CD96E03C138EC4C9EF3F5E78ACE81FA01D34A801C9AA212DE45049B5DDE65DEECEC263
C94804976ECE8A0BB120902CFB02CFE6512313A3

F33AE998463DC0AE6EC378E8DD5E2B86207C5D89873B8522CB7FCA6AF4410A85
FAD3BDAC07EE025BC0AF3B802642724CB40717BD7C96F4C6BE0164AAB959C12D
EE50B5E32EAD50A0E90250845FC7D96C9E7E20AA09330E7A16A874477EACFE468887F82BB097BB6C79C12373
21DE8F7559F7A148CAE20E9B2295127ABD9A1E61
```

- ¿Qué ha pasado con la clave privada d?
- Genere de forma automática una clave de 1.024 bits.
- Copie el valor de la clave privada d al portapapeles, péguela en la ventana de la clave pública e y genere esa clave ahora manualmente.
- Repita los dos pasos anteriores para una clave de 2.048 bits.
- ¿Serían débiles estas claves?

Ejercicio 5. Claves privadas parejas

Con el software genRSA y genere manualmente estas tres claves y observe qué sucede con las claves privadas parejas que indica el programa.

Clave decimal de 14 bits: p = 101; q = 101; e = 37.

Clave decimal de 30 bits: p = 24989; q = 24989; e = 31.

Clave hexadecimal de 44 bits: p = 34301B; q = 34301B; e = 5701.

Nota: En este último caso, el programa tardará varios minutos en entregar la respuesta, indicando además durante su ejecución que la aplicación "No responde". No haga caso de ello; es debido a que existirán 3.420.187 (valor decimal de 34301B) claves privadas parejas y debe escribir todos esos valores en la ventana correspondiente, como se muestra en las siguientes figuras.

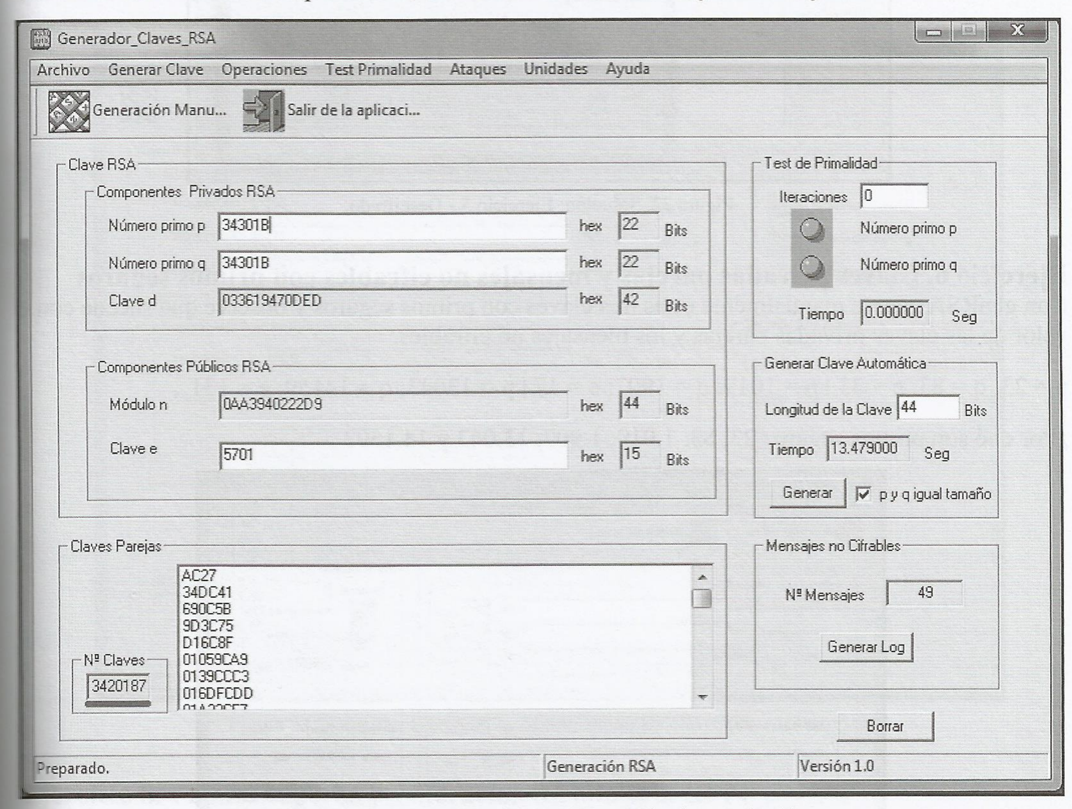


Figura 31. Solución: Ejercicio 5 - Claves privadas parejas (1).

Sin contar con la ayuda de ningún software, excepto una calculadora, y usando las ecuaciones que se han explicado encuentre la cantidad y los valores de las claves privadas parejas de esta clave RSA de 20 bits: Clave RSA: p = 941, q = 853, e = 121.

Hecho esto, compruebe con el software genRSA que se descifra el criptograma con cualquiera de las claves privadas.

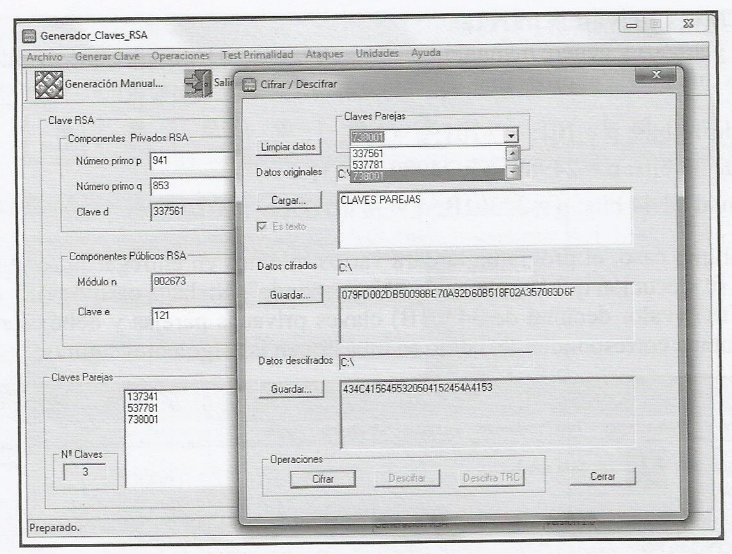


Figura 32. Solución: Ejercicio 5 - Descifrado.

Ejercicio 6. Claves privadas parejas y mensajes no cifrables con primos seguros

Con genRSA genere manualmente estas tres claves con primos seguros y observe qué sucede con el valor de las claves privadas parejas y los mensajes no cifrables.

p = 23; q = 83; e = 31 | p = 1019; q = 1907; e = 17 | p = 13043; q = 14159; e = 131.

¿Por qué son primos seguros 23, 83, 1.019, 1.907, 13.043 y 14.159?

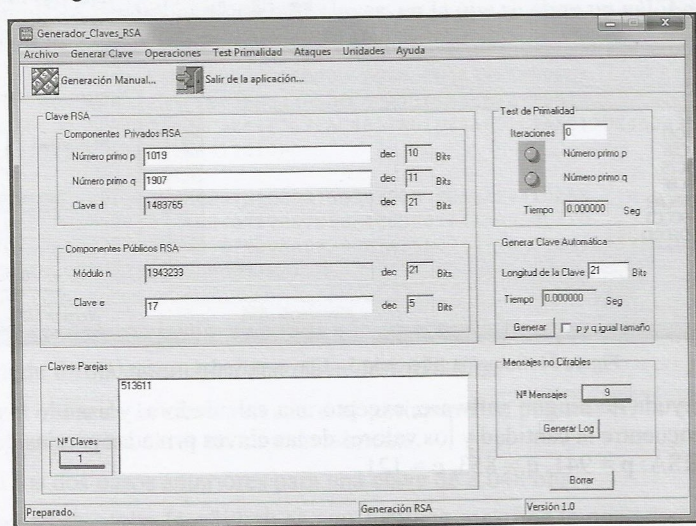


Figura 33. Solución: Ejercicio 6 - Claves privadas parejas y mensajes no cifrables con primos seguros.

Ejercicio 7. Minimizando las claves privadas parejas

Con el software genRSA encuentre todas las claves RSA posibles para los primos dados p = 53, q = 79, en función de la clave pública elegida e desde su valor más pequeño posible hasta un número menor que 100.

Solución:

Como n = 53 x 79 = 4.187 y φ(n) = (53-1)(79-1) = 4.056 = 23 x 3 x 132, los valores posibles de la clave pública e menores que 100 serán: 5, 7, 11, 17, 19, 23, 25, 29, 31, 35, 37, 41, 43, 47, 49, 53, 55, 59, 61, 67, 71, 73, 77, 79, 83, 85, 89, 95, 97.

Genere todas estas claves y observe los valores de las claves privadas parejas. ¿Qué otras claves públicas e entregan 25 CPP?

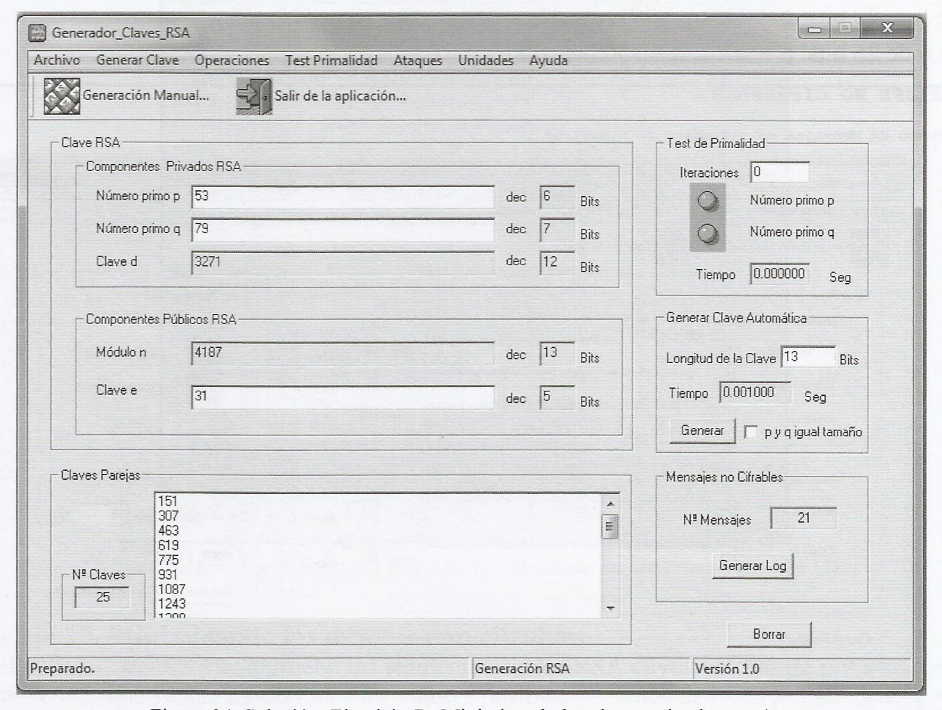


Figura 34. Solución: Ejercicio 7 - Minimizando las claves privadas parejas.

Ejercicio 8. Primos seguros y generación de una sola CPPP

Con el software genRSA compruebe que la elección de primos seguros no es sinónimo de generación de claves con una sola CPPP, si bien su valor siempre será muy bajo.

Genere una clave RSA con los primos seguros p = 11, q = 47 y los siguientes valores de la clave pública e = 3, 7, 9, 11, 13, 17, 19, 21, 27, 29, 31, 33, 37, 39, 41, 43, 47 y 49.

¿Todas las claves han salido con una sola clave privada pareja?

Ejercicio 9. Calculando los números no cifrables

Con el software genRSA y luego con ExpoCrip genere estas 6 claves RSA, apunte la cantidad de mensajes o números nos cifrables y encuentre esos valores haciendo la operación que le permite la aplicación. Dado el alto tiempo que toma esta operación para números grandes, solamente se generarán claves de hasta 25 bits.

Solución:

Clave 1 de 8 bits: p = 13, q = 17, e = 7

Clave 2 de 12 bits: p = 47, q = 59, e = 19

Clave 3 de 15 bits: p = 131, q = 149, e = 41

Clave 4 de 18 bits: p = 409, q = 499, e = 31

Clave 5 de 20 bits: p = 743, q = 991, e = 31

Clave 6 de 25 bits: p = 3061, q = 7603, e =127

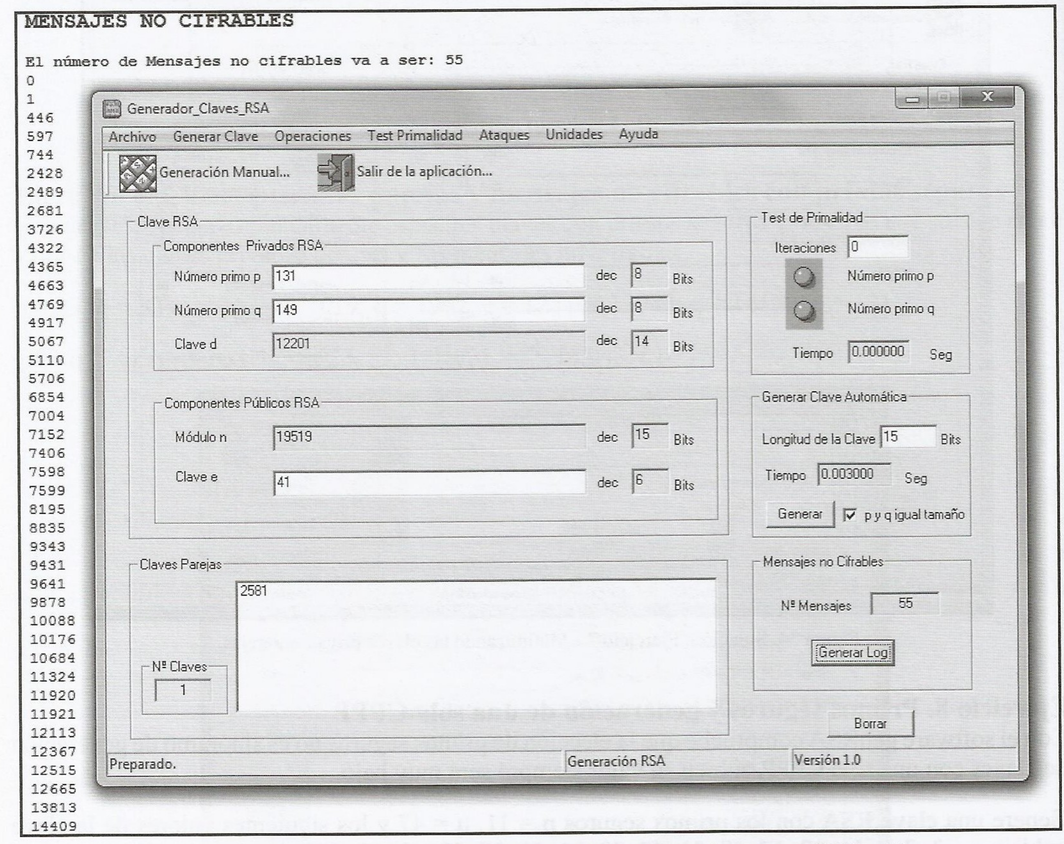


Figura 35. Ejercicio 9. Solución: Ejercicio 9 - Calculando los números no cifrables (1).

Ahora contando solamente con el software Fortaleza de Cifrados, encuentre los 49 números no cifrables de la clave RSA con p = 409, q = 499, e = 31.

Solución:

$$\sigma n = [1 + \text{mcd}\ (e-1, p-1)][1 + \text{mcd}\ (e-1, q-1)]$$

Los números no cifrables serán:

$N = [q\{inv\ (q, p)\}Np + p\{inv\ (p, q)\}Nq] \bmod n$

con:

Np las soluciones de $N^e \bmod p = N$

Nq las soluciones de $N^e \bmod q = N$

```
CLAVE GENERADA

Número primo P generado: 409
Número primo Q generado: 499
Componente (módulo n) generado: 204091
Componente d privado generado: 19663
Componente e generado: 31

MENSAJES NO CIFRABLES

El número de Mensajes no cifrables va a ser: 49
```

0	49899	96168	121117	155828
1	56388	97804	121118	177505
1636	58024	97805	122753	179141
1637	72855	99440	122754	179142
23313	74491	99441	124390	180777
23314	79701	104650	129600	180778
24949	81337	104651	131236	202454
24950	81338	106286	146067	202455
26586	82973	106287	147703	204090
48263	82974	107923	154192	

Figura 36. Ejercicio 9. Solución: Ejercicio 9 - Calculando los números no cifrables (2).

Ejercicio 10. Minimizando los números no cifrables

Con el software genRSA compruebe las siguientes claves RSA cuyos valores de p y q son primos seguros, en las que se han usado todos los valores posibles y válidos de la clave pública e.

Clave RSA: p = 5, q = 7 Valores de clave pública: 2 < e < 35

Clave RSA: p = 5, q = 11 Valores de clave pública: 2 < e < 55

Clave RSA: p = 7, q = 11 Valores de clave pública: 2 < e < 77

Clave RSA: p = 5, q = 23 Valores de clave pública: 2 < e < 115

Clave RSA: p = 23, q = 59 Valores de clave pública: 2 < e < 1.357

¿Qué sucede con la cantidad y proporción de valores de la clave pública e que dan como resultado sólo 9 números no cifrables a medida que aumentan de tamaño los valores de p y q?

Ejemplo:

Clave RSA: p = 7, q = 11
p = 7 = (2r + 1) = (2*3 + 1) con r = 3
q = 11 = (2s + 1) = (2*5 + 1) con s = 5
NNC = (1 + 2)(1 + 2) = 9; e = 17, 23, 29, 47, 53, 59
NNC = 3(2*r + 1) = 3(2*3 + 1) = 21; e = 7, 13, 19, 37, 43, 49
NNC = 3(2*s + 1) = 3(2*5 + 1) = 33; e = 11, 41
NNC = (2*r + 1) (2*s + 1) = (2*3 + 1) (2*5 + 1) = 77; e = 31

Genere, a continuación, con el software genRSA claves en las que la clave pública e = φ(n)/2 + 1 y observe qué sucede con la cantidad de números no cifrables σ(n). En todo caso, compruebe que se da esa relación entre e y φ(n).

Clave de 20 bits: p = 937, q = 991, e = 463321
Clave de 22 bits: p = 1493, q = 1499, e = 1117509
Clave de 24 bits: p = 3863, q = 4007, e = 7735587
Clave de 26 bits: p = 7499, q = 7057, e = 26452945
Clave de 28 bits: p = 11743, q = 14731, e = 86479831
Clave de 30 bits: p = 26209, q = 28447, e = 372756385
Clave de 32 bits: p = 60343, q = 61091, e = 1843146391
Clave de 64 bits: p = E2BE51B1, q = BAABE269, e = 52AB428148FF47C1

Observe en este último caso el error que entrega el programa al indicar el Número de Mensajes no Cifrables. ¿Cuál es el valor verdadero? Para ello use el programa ExpoCrip introduciendo en los valores de p, q y e en formato decimal.

Ejercicio 11. ¿Hay que preocuparse por estos números no cifrables?

Con el software genRSA genere estas tres claves de tamaños estándar y observe la cantidad de números no cifrables σ(n).

Clave 1:

```
1CC0C6A8909BE5B660807C6156F93DDDE67D4F08877AFD4C0752B0C721A31AA951CCB5DEE7AD6CEE5BAE9721
E0022DE63B2D3575D6C37EB410739057D845E24C7C0106F90C2DB8C3537D9ED1226721C4175DFC37F077FC95
FF1211DBD9BEA59F4F572C529F71FDC2EBF960E2E91AE8CEDD5E030AC416CF0AD2FB2FF81F151FB1E110001
```

Clave 2:

```
CB71C1D96D4B530774C47D0155E9EE4ED48514326A38DE6D2FC9C525FEBD0D20D329483677C41D06604047E9
D2DBE0244B31E0621E54E1615E1E25EBFDC6D961DCB620812EA9E4A0D4F48B4D214B0C78DCA78D7C81E978E9
EFC0D08661CCC6969F04875B765559D9658BE6DC9975D73C1C44A5604935AC0183708230262F874110001
```

Clave 3:

```
BF11FD9D42B665A1C2D37194E99F8EFBBA5C04D0D47F86CAA41107DFDA55D00819ABD044F3B433DF14B736F4
BD443931751715DAFB3EB1107E7FD05410F42D4759F8CAB3884785CF8F60ECBF902E525B92E82331793674CD
3595DE3839FC8DBE2FC094F4EAB583FA9C963CFE49A602017B98D3DE38ACF888311CDA933CB61801D4489927
2991BEF647DA3AC06F10D57FC68E0813B8E7B52D48DF166DA4BAF4C7FDBA192B95E307651A01EA4232ECCCF4
570B08B7E51701B5712113275DD7CABEA90352410C542F7744612E47AAEB9C1E0962B591B0AB5F44C621190A
DEDA2E4DCB2B2EB1B66E9EF46E02EF19FE94ADC5859DF9E9422C39A1EFE60E5C504B89D910001
```

Ejercicio 12. Aplicación del TRC en el descifrado RSA

Para las siguientes claves, descifre el criptograma C utilizando el teorema chino del resto.

Clave RSA1: p = 223, q = 251, e = 131, C = 44.683
Clave RSA2: p = 23.357, q = 29.759, e = 4321, C = 487.735.883
Clave RSA3: p = 18EE7F9, q = 1DB117F, e = 5FB9, C = 2E1EEB6C98D7B (M en decimal)

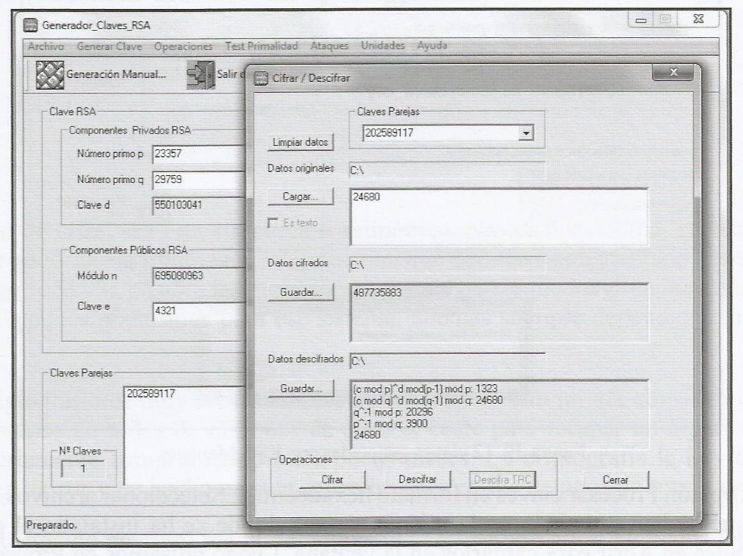


Figura 37. Solución: Ejercicio 12 - Aplicación del TRC en el descifrado RSA.

Ejercicio 13. Generación de múltiples claves RSA con RSA MANAGER

Mediante el software RSA Manager disponible gratuitamente en la red CRIPTORED[4] es sencillo generar automáticamente varias claves RSA.

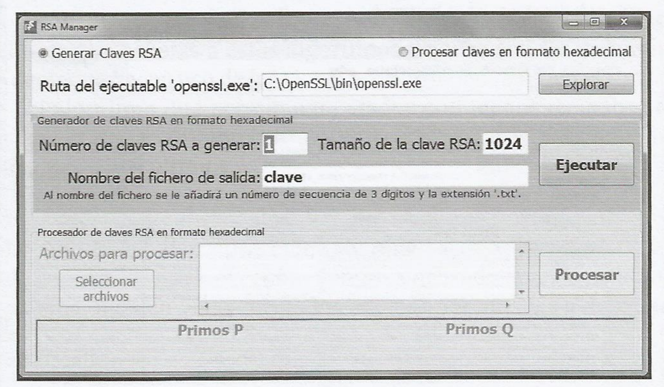


Figura 38. Pantalla del programa RSA Manager.

Con la opción por defecto Generar Claves RSA, una vez indicada la ruta donde está instalado OpenSSL, el número de claves a generar, su tamaño y el nombre genérico de estas claves, se pulsa en Ejecutar.

4 *http://www.criptored.upm.es/software/sw_m001n.htm*

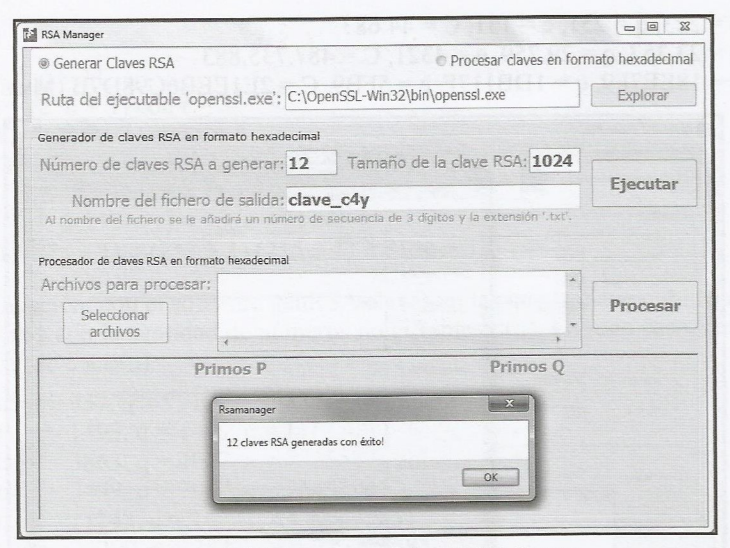


Figura 39. 12 claves generadas con RSA Manager.

Ahora se usa la opción Procesar claves en formato hexadecimal, Seleccionar archivos, se seleccionan las 12 claves que se han guardado en la misma carpeta donde se ha instalado el programa RSA Manager y se pulsa en Abrir para cargarlos en la ventana. Luego pulsamos en Procesar, obteniendo la siguiente pantalla.

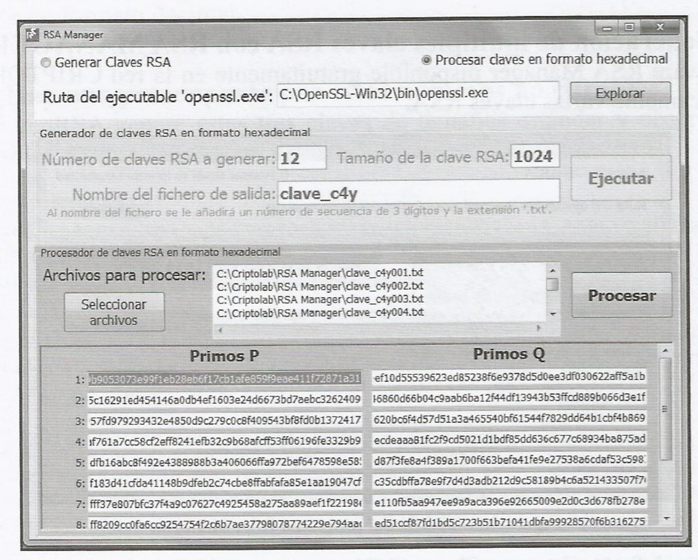


Figura 40. RSA Manager y claves en formato texto.

Haciendo un solo clic en un valor de p y de q para seleccionarlo (otro clic para deseleccionar) puede copiar ese valor al portapapeles y copiarlo luego en el programa genRSA y estudiar otra serie de parámetros.

Capítulo IV
La seguridad de la criptografía de clave pública y el algoritmo RSA

Todo algoritmo de cifra, sea ésta simétrica o asimétrica, clásica o moderna, en bloque o en flujo, tendrá siempre su fortaleza limitada al mejor ataque que se haya implementado hasta ese momento. Es decir, la seguridad y la posible fecha de caducidad del algoritmo vendrán marcadas por el estado del arte de los ataques planteados para ese sistema, además siempre dependiente del avance de la tecnología.

Así que en criptografía es muy recomendable ser cautos; lo que un día consideramos como muy seguro y estimamos en miles de millones de años el coste para romper un algoritmo o una cifra, podría estar seriamente comprometido a los pocos meses. En criptografía la Ley de *Moore* no se cumple, pueden existir y de hecho existen saltos bruscos en los tiempos de rotura de sistemas.

Un caso muy conocido en 1999 fue el del algoritmo *Data Encryption Standard* DES, un estándar mundial de la cifra simétrica en aquellos años y usado entre otros en SSL, que tras cuatro intentos de ataque distribuido en red en dos años denominados DES Challenge, y precisamente promovidos y organizados por RSA Laboratories, al final sucumbió en menos de 24 horas, simplemente porque su longitud de clave de 56 bits era la adecuada para los años 70 y 80, pero no ya para mediados de los 90.

Aunque el anterior fue un ataque por fuerza bruta, poco elegante, lo cierto es que uno de los ataques más buscados por los criptoanalistas a estos algoritmos de cifra son los denominados precisamente criptoanalíticos, es decir que hacen uso de las técnicas de criptoanálisis, en el fondo buscar debilidades en la construcción del algoritmo que permitan un ataque con menos recursos computacionales que los que necesita la fuerza bruta, en tanto las claves actuales son lo suficientemente grandes como para hacer impracticables estos últimos.

Los sistemas asimétricos, y particularmente RSA que es el que nos interesa en este libro, corren la misma suerte. De hecho, RSA Laboratories durante muchos años mantuvo desafíos para factorizar diferentes valores de n en sus dos primos como veremos más adelante. Y no sólo eso, los algoritmos como parte de un sistema, pueden verse afectados además por ataques a dicho sistema.

Veamos todo esto en detalle.

4.1. Ataques criptoanalíticos al algoritmo RSA

En los últimos 20 años se han documentado multitud de ataques criptoanalíticos por software o hardware para anular la seguridad del algoritmo RSA.

En este capítulo se pretende dar un vistazo a los ataques más significativos por su utilidad docente y porque quizás pudieran servir de referencia para nuevos ataques a este algoritmo o similares. Adicionalmente a los expuestos existen toda una serie de ataques que si bien no se centran en el algoritmo RSA pueden simplificar la deducción de la clave privada o la rotura del algoritmo; ejemplos de ellos son ataques basados en el análisis de diferencias temporales en la generación de criptogramas, el estudio del consumo energético, la manipulación eléctrica de los dispositivos, ataques tipo BPA *(Branch Prediction Analysis), etc. En Internet y en la bibliografía de este libro puede encontrar una amplia documentación sobre ellos.*

4.1.1. El problema de la factorización entera

La seguridad del algoritmo RSA se basa en la dificultad computacional que conlleva encontrar los dos factores primos de un número compuesto muy grande, el módulo n. Es posible que el lector haya leído esta afirmación en multitud de sitios, si bien es cierto en el caso del algoritmo RSA no se ha demostrado que su seguridad equivalga precisamente a resolver este reto y no existan otros procedimientos a través de los cuales atacar RSA sea más rápido. En cualquier caso, a falta de formalización a este respecto es común hablar de la seguridad de RSA vinculada a la factorización de su módulo n. Como verá en este capítulo, existen muchas otras formas de atacar al algoritmo.

Precisamente encontrar los dos factores primos de un número compuesto, para nosotros el módulo n, es lo que se conoce como el problema de la factorización entera, uno de los problemas denominados No Polinomiales o de tipo NP, muy usados en criptografía. Se trata de un problema en el que en un sentido el cálculo es muy fácil y rápido (por ejemplo multiplicar dos números primos) pero que en sentido contrario (por ejemplo, encontrar esos dos factores conociendo el producto) se vuelve computacionalmente intratable a medida que la entrada es cada vez mayor. Es decir, requiere de unos recursos informáticos excesivos y, por tanto, de un tiempo de cálculo exorbitante.

Aunque no sea matemáticamente exacto, se podría aceptar que la primera operación es *lineal* en el sentido que la cantidad de operaciones a realizar es directamente proporcional al tamaño de los dos operandos; en cambio, la segunda operación es de tipo *exponencial* de manera que, por ejemplo, aumentando al doble el tamaño de la entrada, el número de cálculos a realizar no aumenta también al doble sino mucho más.

Rivest, *Shamir* y *Adleman* usaron el problema de la factorización entera, el tamaño del módulo n, como elemento de fortaleza en el diseño de su algoritmo RSA. Aunque este principio se mantiene en 2013 se recomienda que los primos que componen n sean de al menos 512 bits (unos 155 dígitos) cuyo producto es un número de 1.024 bits (unos 310 dígitos), aunque lo ideal sería migrar ya a claves de mínimo 2.048 bits producto de dos primos de 1.024 bits como ya lo hace, por ejemplo, la banca online en sus certificados digitales X.509.

En las últimas décadas se han propuesto diferentes algoritmos de factorización entera para atacar a RSA. Estos algoritmos pueden dividirse en dos grupos, los denominados de propósito general y los de propósito específico. Entre los más conocidos se encuentran:

1) Método de factorización directa o criba de Eratóstenes

2) Método de *Fermat*

3) Método de *Euler*

4) Método de Dixon

5) *Williams* p+1 Factorization Method

6) Método de Pollar rho

7) Método de Pollar p-1

8) Método de las fracciones continuas

9) Método de las curvas elípticas

10) Método de la criba numérica

En el trabajo de *Connely Barnes* de la Universidad de Oregón de diciembre de 2004, *Integer Factorization Algorithms*, se puede observar un interesante estudio comparativo entre algunos de esos algoritmos: método de factorización directa, método de *Fermat*, método de *Pollard* rho, método de *Brent* y método de *Pollard* p-1.

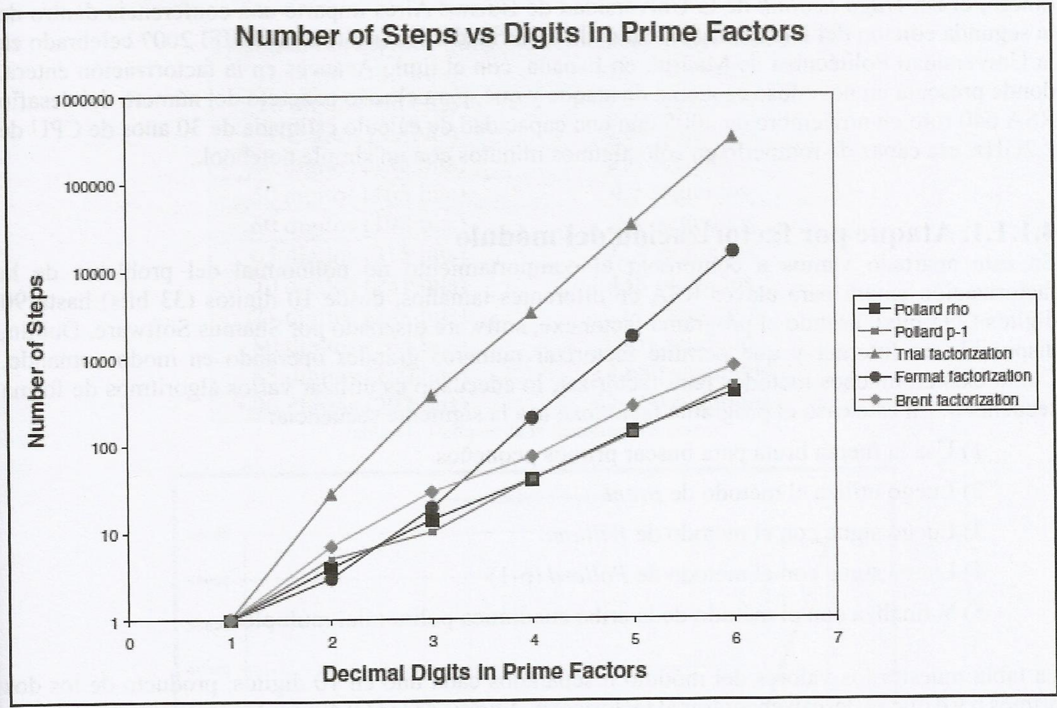


Figura 41. Número de pasos según el algoritmo de factorización en escala logarítmica.

El mejor algoritmo conocido a la fecha, el de la criba numérica o *General Number Field Sieve* GNFS, tiene asociada una complejidad representada en la siguiente expresión para un número de b bits:

$$O\left(\exp\left(\left(\frac{64}{9}b\right)^{\frac{1}{3}}(\log b)^{\frac{2}{3}}\right)\right)$$

Como se puede observar, el factor b que representa en bits el tamaño del número a factorizar (n), se encuentra como un elemento del exponente y por ello el carácter exponencial de este problema NP.

El problema de la factorización entera es uno de esos problemas en matemáticas clasificados como abiertos; es decir, siempre puede aparecer un nuevo algoritmo que mejore las prestaciones de los anteriores o bien que presente ciertas ventajas frente a aquellos en casos especiales. Lo que es cierto es que a la fecha nadie ha logrado quitarle esa característica de problema NP o comportamiento exponencial. Dos ejemplos de los últimos avances en la factorización entera son el último número factorizado en el RSA *Factoring Challenge* con 768 bits y el trabajo del Dr. *Hugo Scolnik*.

En el año 2007 RSA da por terminado ese desafío, poco después de que en el año 2005 se factorizasen números de 193 dígitos (640 bits) y 200 dígitos (663 bits). Ya fuera de concurso a estos premios en metálico, el 12 diciembre de 2009 un equipo de seis instituciones de investigación lideradas por *Thorsten Kleinjung* logra factorizar el mayor número RSA hasta el momento: 768 bits. En aquel año en que RSA anuncia que no sigue activo ese desafío en cuanto a la entrega de premios en dinero, el Dr. *Hugo Scolnik* de la Universidad de Buenos Aires imparte una conferencia dentro de la segunda edición del Día Internacional de la Seguridad de la Información DISI 2007 celebrado en la Universidad Politécnica de Madrid, en España, con el título Avances en la factorización entera, donde presenta un novedoso esquema de ataque y que, para el caso concreto del número del desafío RSA 640 roto en noviembre de 2005 con una capacidad de cálculo estimada de 30 años de CPU de 2.2GHz, era capaz de romperlo en sólo algunos minutos con un simple notebook.

4.1.1.1. Ataque por factorización del módulo

En este apartado vamos a comprobar el comportamiento no polinomial del problema de la factorización entera para claves RSA de diferentes tamaños, desde 10 dígitos (33 bits) hasta 90 dígitos (297 bits), usando el programa factor.exe, software diseñado por Shamus Software, Dublin, disponible en Internet y que permite factorizar números grandes operando en modo comando. Como existen muchos métodos para factorizar, lo adecuado es utilizar varios algoritmos de forma secuencial. En este caso el programa factor.exe usa la siguiente secuencia:

1) Usa la fuerza bruta para buscar primos pequeños
2) Luego utiliza el método de *Brent*
3) Luego sigue con el método de *William*
4) Luego sigue con el método de *Pollard* (p-1)
5) Y finaliza con el método de la criba cuadrática polinomial múltiple

La tabla muestra los valores del módulo n separados cada uno en 10 dígitos, producto de los dos primos p y q que se desean encontrar al factorizar n. A partir de los 60 dígitos la separación se realizará de cinco en cinco dígitos pues ya es muy significativo el comportamiento de una curva exponencial. En la sección de bibliografía puede acceder al documento completo de esa investigación.

Valores del cuerpo n = p * q que se desea factorizar, en formato decimal	
10 dígitos	6384329603
20 dígitos	37898577272681469353
30 dígitos	461385432750893547024930873479

Valores del cuerpo n = p * q que se desea factorizar, en formato decimal	
40 dígitos	2854234617331252680802693649120361233813
50 dígitos	35421923062244554951243462450002521261335776714419
60 dígitos	437610495647307157001072669764047918009218116295581647528901
65 dígitos	11640646193039951929832971988054197479161937220812816707953189181
70 dígitos	3583789913020477384239508976983261788340229625300830452087617770773939
75 dígitos	899860202281103733904853835097291544406726946866166258013445125433485561231
80 dígitos	65666185406214674912036156868849798957525212130336139657926553289114114640370373
85 dígitos	1230230645604630726004851947131604313160468353546777397194993013659679459817729459009
90 dígitos	249466747125192798859465494967943591682682693901288852594696933237244570457253508108210113

Tabla 10. Valores decimales de módulo n a factorizar.

Introduciendo estos valores en el software factor.exe, y con una máquina PC Hewlett-Packard con CPU Intel Core i7-2600 CPU 3.40 GHz, una RAM de 8.0 GB y sistema operativo de 64 bits, se obtienen la siguiente tabla y gráfica.

Tiempos requeridos para la factorización del módulo n	
10 dígitos (33 bits)	0,01 segundos
20 dígitos (66 bits)	0,03 segundos
30 dígitos (99 bits)	0,05 segundos
40 dígitos (132 bits)	1,4 segundos
50 dígitos (165 bits)	4,3 segundos
60 dígitos (198 bits)	25,7 segundos
65 dígitos (215 bits)	92,4 segundos
70 dígitos (231 bits)	438 segundos
75 dígitos (248 bits)	1.723 segundos
80 dígitos (264 bits)	5.461 segundos
85 dígitos (280 bits)	14.680 segundos
90 dígitos (297 bits)	67.150 segundos

Tabla 11. Tiempos de la factorización entera.

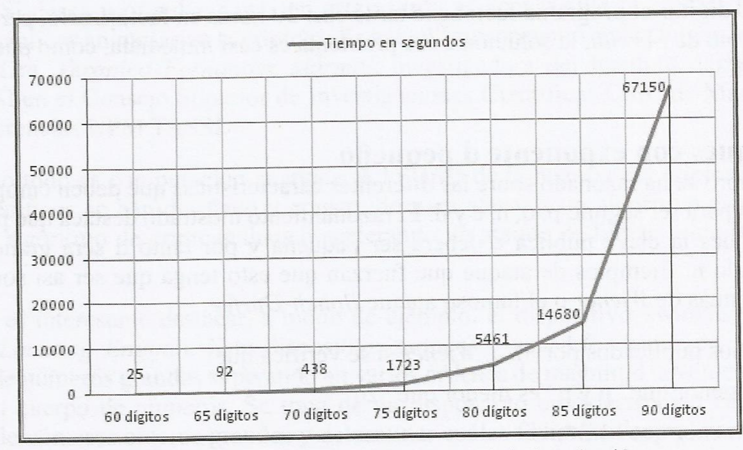


Figura 42. Tiempos asociados al problema de la factorización.

Observe que el último valor de 67.150 segundos corresponde a un tiempo igual a 18 horas, 39 minutos y 10 segundos. En la siguiente figura se muestra una captura de pantalla cuando el programa entrega el valor de p y q pasado ese tiempo.

```
Símbolo del sistema
         phase 2 - trying last prime less than 5000000
curve 297 phase 1 - trying all primes less than 50000
         phase 2 - trying last prime less than 5000000
curve 298 phase 1 - trying all primes less than 50000
         phase 2 - trying last prime less than 5000000
curve 299 phase 1 - trying all primes less than 50000
         phase 2 - trying last prime less than 5000000
curve 300 phase 1 - trying all primes less than 50000
         phase 2 - trying last prime less than 5000000
finally - the multiple polynomial quadratic sieve - with large prime (*)
using multiplier k= 2 and 16018 small primes as factor base
working... 15852
trying...
working...*15852
trying...
PRIME FACTOR     499466462462889292700096245572889152567727729
PRIME FACTOR     499466462462889292700096245572889152567727697
End     = 16:10:11,66
Start   = 21:31:01,63
Hours   = 18
Minutes = 39
Seconds = 10
Total   = 67150

C:\Criptolab\Factor>_
```

Figura 43. Captura de pantalla del programa factor.exe para una clave de 90 dígitos.

En el ejemplo anterior los primos seleccionados eran demasiado cercanos ya que como podrá apreciar sólo cambian los últimos 3 dígitos; de hecho, son primos consecutivos. En RSA se recomienda que los primos p y q estén lo suficientemente separados.

¿Qué sucedería si elegimos primos p y q muy cercanos? La respuesta es que en este caso el algoritmo de *Fermat* sería muy eficiente porque resuelve la factorización en muy pocos pasos. Y es precisamente lo que ha sucedido en el ejemplo anterior en que factorizamos números compuestos de 10 hasta 90 dígitos: los valores de p y q eran precisamente primos consecutivos. Para una clave RSA de módulo n 90 dígitos el programa factor.exe tarda casi 20 horas en factorizarlo, pero si se hubiese usado el algoritmo de *Fermat*, la solución a ese problema es casi inmediata, como puede comprobar en el apartado 4.3.

4.1.1.2. Ataques con exponente d pequeño

A lo largo del libro se ha razonado sobre las diferentes características que deben cumplir los valores de la clave RSA para ser segura: p, q, n, e y d. El razonamiento mostrado destaca que para minimizar numerosos ataques la clave pública e deberá ser pequeña y por tanto d será grande, de tamaño similar al módulo n. Ejemplos de ataque que fuerzan que esto tenga que ser así son el ataque de fracciones continuas de *Wiener* o el famoso ataque *Boneh-Durfee*.

Según los estudios publicados por *M. J. Wiener* si se verifica que:

a) q “es menor que” p y p “es menor que” 2q

b) d “es menor que” $1/3*n^{1/4}$

Entonces existe un algoritmo que corre en tiempo polinómico O(log n) y que genera una lista de longitud (log n) de candidatos para d, uno de los cuales es d. Es decir, si la longitud en bits de la clave privada d es menor que la cuarta parte de la longitud en bits de n, d puede ser determinado en tiempo polinómico, es decir, romper el algoritmo.

El ataque de *Boneh-Durfee* mejora la capacidad del primero y demuestra que es posible romper RSA si $d < n^{0.292}$.

4.1.1.3. Computación cuántica y dispositivos específicos. Twinkle

En los últimos años la ciencia de la computación cuántica ha tenido un gran interés, especialmente desde un punto de vista analítico. Las matemáticas y la física demuestran que es posible construir un artefacto que haga uso de la siempre esquiva física cuántica para crear ordenadores de una gran potencia. Tanto que haría que muchos de los problemas matemáticos intratables hoy día, problemas NP, encontrarían solución. Un ejemplo de ello es el problema de la factorización entera. Diferentes propuestas han resaltado algoritmos para solucionar el problema de manera eficiente (si existieran ordenadores cuánticos) y por tanto romper la seguridad de algoritmos como RSA. Un ejemplo de ello es el algoritmo de Shor, un algoritmo cuántico que permite, en teoría, descomponer en factores un número N en tiempo $\underline{\mathbf{O}}((\log N)^3)$, es decir, polinómico en lugar de exponencial.

Si entendemos los procedimientos para construir ordenadores de este tipo y conocemos algoritmos con utilidad en criptoanálisis, entonces ¿por qué RSA no está roto? Ante esta pregunta existen múltiples respuestas posibles, desde las más puras conspiratorias y paranoicas (*conspiranoicas*) que nos dicen que la NSA ya tiene este tipo de dispositivos y puede "*leernos hasta la mente*" o las más sensatas, basadas en las publicaciones científicas serias, en las que es fácil observar que todavía existen problemas de ingeniería serios a resolver. Es una cuestión de tiempo y dinero. De todas maneras, el lector no debe preocuparse; existen muchas otras maneras de proteger comunicaciones digitales sin que la computación cuántica suponga un problema.

En la actualidad, puede leer noticias de todo tipo de empresas que están vendiendo ordenadores cuánticos a medida (específicos) como por ejemplo la empresa D-ware; ante estos datos se debe ser cauteloso. Si desea profundizar más en este asunto es aconsejable la conferencia "Cómo los ordenadores cuánticos aniquilarían la criptografía actual" que presentó en la Universidad Politécnica de Madrid la Dra. *Verónica Fernández Mármol*, investigadora del Instituto de Seguridad de la Información ISI en el Consejo Superior de Investigaciones Científicas CSIC de Madrid, en el VIII Ciclo de Conferencias UPM TASSI.

No obstante, no todo es computación cuántica. A lo largo de la historia del criptoanálisis diversos artefactos específicos se han diseñado y construido para atacar a un algoritmo específico, ya sea acelerando un algoritmo de criptoanálisis o acelerando un ataque de fuerza bruta (Colossus, DES Cracker).

En este punto es interesante destacar, a modo de ejemplo, el dispositivo Twinkle *(The Weizmann Institute Key Locating Engine*). Este dispositivo propuesto por *Adi Shamir* en 1999 mejora la factorización de números grandes superando en varios órdenes de magnitud la velocidad del método de la criba del cuerpo de números. Se trata de un dispositivo optoelectrónico capaz de analizar 100 millones de números enteros grandes y determinar cuáles factorizan completamente sobre una

base de factores formada por los 200.000 primeros números primos, todo esto en menos de 10 milisegundos.

El coste de este dispositivo era similar al de un potente PC o estación de trabajo, y era entre 500 y 1000 veces más rápido que el método de la criba cuadrática en la etapa de la criba. Las ventajas de este dispositivo son claras en lo referente a la fase de la criba, pero eso no supone que la recuperación de la clave sea más sencilla. El problema de esta propuesta tiene que ver con su escalabilidad cuando el tamaño del módulo crece. Para el caso de un módulo de 1.024 bits harían falta 45.000 dispositivos de este tipo y unos 500.000 años para realizar la criba.

Robert D. Silverman en su artículo de 1999 en RSA Laboratories, *An Analysis of Shamir's Factoring Device*, lo deja bastante claro: "*the idea presented by Dr. Shamir is a nice theoretical advance, but until it can be implemented and the matrix difficulties resolved it will not be a threat to even 768-bit RSA keys, let alone 1024*".

Si bien es cierto, el tema todavía no está cerrado. Diferentes propuestas en este sentido se siguen publicado, por ejemplo TWIRL.

4.1.2. Ataque por cifrado cíclico

Este ataque se basa en la posibilidad de descifrar un criptograma usando la misma clave de cifra, es decir la clave pública, mediante un ataque que utiliza sólo datos de la víctima y que son públicos. Otro tema es que esto implique mucho tiempo de cómputo para claves de tamaños actuales sobre los mil bits. Con ello, no romperemos la clave privada del destino pero sí el secreto o confidencialidad que se esperaba lograr con esa cifra.

La cuestión es la siguiente:

Como $C = N^e \bmod n$, con N un valor secreto, se realizan cifrados sucesivos de los criptogramas C_i resultantes con la misma clave pública e. Si en uno de estos cifrados se obtiene nuevamente el cifrado C original con el que se ha iniciado el ataque, resulta obvio que el valor del paso anterior será el secreto N buscado. Esto se debe a que RSA es un grupo multiplicativo. Para dificultar este tipo de ataques, es interesante usar primos seguros de forma que los subgrupos de trabajo sean lo suficientemente altos.

Por tanto, conocido o capturado el criptograma C, se hacen los siguientes cifrados:

$C_i = C^e_{i-1} \bmod n$; para i = 1, 2, ..., con $C_0 = C$

Si en el cifrado i-ésimo se encuentra el criptograma C inicial, entonces el cifrado anterior (i-1) será el número secreto. Veámoslo con un ejemplo.

Se captura un criptograma C = 50 en el que se sabe va enmascarada una clave secreta K que ha enviado Alicia a Bernardo, de quien sólo se conocen sus claves públicas n = 187 y e = 7. Aunque para este caso sería elemental encontrar los primos p y q factorizando el módulo 187 = 11 x 17, se verá qué sucede haciendo un ataque por cifrado cíclico.

1) Primera operación: $C^e \bmod n = 50^7 \bmod 187 = 118$

2) Segunda operación: $C^e \bmod n = 118^7 \bmod 187 = 101$

3) Tercera operación: C^e mod n = 101^7 mod 187= 84

4) Cuarta operación: C^e mod n = 84^7 mod 187= 50

Como en la cuarta operación se ha llegado al valor 50 con el que partió el ataque, resulta claro que la clave secreta era K = 84.

En este ejemplo se han realizado tan sólo 4 operaciones para romper el secreto. El número de operaciones necesarias hasta encontrar el secreto va a depender del valor con el que se comience el ataque (criptograma capturado). ¿Qué sucedería si, por ejemplo, el valor del criptograma inicial fuese 100 en vez de 50?

1) Primera operación: C^e mod n = 100^7 mod 187 = 144

2) Segunda operación: C^e mod n = 144^7 mod 187 = 100

En este caso sólo se necesitarían 2 operaciones para romper el secreto K = 144.

Este es un fenómeno muy interesante, la formación de diversos anillos de cifrados hasta romper el secreto en función del criptograma con el cual se inicia el ataque pero que no abordaremos en este libro; si desea puede consultarlo en la Lección 9 del MOOC Crypt4you El algoritmo RSA.

El problema de este tipo de ataque es que, además de necesitar como dato el criptograma, no resulta sencillo llevarlo a un escenario de ataque distribuido pues los subgrupos que se forman son muy grandes cuando aumenta el tamaño de la clave. Existe una proporcionalidad entre el tamaño del módulo n y el tamaño de la ventana de cifrados del ataque, con lo cual va a ser muchísimo más difícil atacar, por ejemplo, una clave de 100 bits que una de 80.

Por otra parte, existe también una gran dispersión en el número de cifrados necesarios para que prospere el ataque ante claves de dimensiones similares. En las siguientes figuras se muestran dos tablas donde se aprecia este efecto. La primera de ellas con el número más repetido y común de vueltas necesarias para que prospere un ataque por cifrado cíclico ante diferentes claves, todas de 24 bits, con diferentes entradas aleatorias del valor C.

Ataque por cifrado cíclico a claves de 24 bits Vuelta en la que prospera el ataque			
n	p	q	Vuelta
11.081.071	3.067	3.613	168
11.433.619	2.969	3.851	260
11.948.269	3.253	3.673	540
11.960.323	3.109	3.847	960
12.026.669	3.929	3.061	2.940
10.474.613	3.449	3.037	4.730
10.539.409	3.187	3.307	7.308
9.885.713	2.917	3.389	13.365
12.296.387	3.019	4.073	127.508
12.314.501	3.019	4.079	511.538
La clave pública es común: e = 421			

Figura 44. Vueltas necesarias en un ataque a claves de 24 bits.

En la siguiente figura, se ataca el mensaje en claro 123 (que da origen al criptograma C con el cual se inicia el ataque, en función de los parámetros de cada clave) también con claves de 24 bits

manteniendo ahora constantes el valor del primo p y de la clave pública e. Se varía sólo el valor del primo q dentro del rango de la clave, en tres casos eligiendo q como primo seguro. Al final de la tabla ambos primos p y q se eligen como primos seguros.

De esta primera figura podría deducirse, siempre en una primera aproximación, que el factor suerte jugará un papel muy importante en este ataque, en tanto los valores de los primos p y q son aleatorios y en el escenario de ataque a una clave de 24 bits de la figura anterior, bien podríamos haber necesitado solamente una centena de cálculos para romper el secreto en la primera clave, como más de medio millón de cálculos en la última.

Ataque por cifrado cíclico a claves de 24 bits Vuelta en que prospera el ataque a M = 123			
n	p	q	Vuelta
9.168.703	3.019	3.037	27.610
9.241.159	3.019	3.061	1.506
9.687.971	3.019	3.209	100.400
10.466.873	3.019	3.467	217.366
10.231.391	3.019	3.389	82.830
10.907.647	3.019	3.613	10.542
11.088.787	3.019	3.673	1.506
11.408.801	3.019	3.779	118.472
11.614.093	3.019	3.847	160.640
11.626.169	3.019	3.851	7.530
12.296.387	3.019	4.073	63.754
12.314.501	3.019	4.079	511.538
10.480.741	3.023	3.467	130.766
11.423.917	3.023	3.779	71.272
12.330.817	3.023	4.079	153.869
3.023, 3.467, 3.779, 4.079 primos seguros			
La clave pública es común: e = 101			

Figura 45. Vueltas para romper el secreto M=123 para diferentes claves de 24 bits.

En este caso se observa una gran dispersión de número de vueltas necesarias para que prospere el ataque con sólo cambiar ligeramente uno de los primos de la clave. No obstante, se aprecia que al utilizar primos seguros en la generación de la clave, existe una clara tendencia a que los espacios de colisión sean mayores que con primos no seguros.

Incidencia de la clave pública e y primos seguros en el ataque

Como este ataque se lleva a cabo realizando sucesivos cifrados con la clave pública e, es decir valor^e mod n, podríamos pensar que ese valor de la clave pública e -que está en el exponente de la operación- va a incidir en la tasa de cifrados y, por tanto, en el rendimiento del ataque. Esto es cierto pero sólo en parte. No resultará significativo ese parámetro porque, como sabemos, la clave pública siempre será un valor relativamente bajo.

La tabla siguiente muestra la tasa de cifrados por segundo que alcanza el programa genRSA en este ataque, en función del tamaño de la clave pública e con valores desde dos hasta cinco dígitos, para una máquina Intel(R) Core(TM) i7-2600 con CPU de 3,40 GHz y Sistema Operativo de 64 bits. No es necesario aumentar más ese tamaño dado que la clave pública estándar es el número 4 de *Fermat* 65.537, de cinco dígitos.

Los valores utilizados en la tabla para la clave pública e son:

e = 23 = 10111 (con 4 bits en 1)

e = 421 = 110100101 (con 5 bits en 1)

e = 6.941 = 1101100011101 (con 8 bits en 1)

e = 37.867 = 1001001111101011 (con 10 bits en 1)

e = 65.537 = 10000000000000001 (con 2 bits en 1)

Clave de 32 bits	Clave pública	Cifrado/segundo
p = 64.693, q = 60.811	e = 23	436
p = 64.693, q = 60.811	e = 421	403
p = 64.693, q = 60.811	e = 6.941	433
p = 64.693, q = 60.811	e =37.867	473
p = 64.693, q = 60.811	e = 65.537	435

Figura 46. Tasa de cifrados por segundo en función del tamaño de clave pública.

La tasa de cifrado por segundo que entrega el programa genRSA es extremadamente baja porque no se puso como condición una alta velocidad de cifra en este software del año 2004. Lógicamente hay implementaciones que multiplican por órdenes de magnitud esta tasa.

Si bien los valores de la clave pública tienen distintos tamaños y un número creciente de bits en 1, que deberían hacer algo más lenta la operación de cifra $C = M^e \bmod n$, lo cierto es que esto no influye en la tasa de cifra de este ataque, debido a que son valores pequeños. Incluso para números grandes este dato de cambio en la tasa de cifrados no es significativo, al menos en este ataque, y puede además tener muchas oscilaciones en función del valor con el que comience el ataque y los valores de la clave, bien en decimal como en hexadecimal. En la siguiente tabla se muestran algunos valores del número de vueltas necesarios para romper el secreto con el programa genRSA, para las mismas claves públicas del ejemplo anterior y para diferentes valores de inicio en el ataque, en este caso M = 12, M = 123, M = 1.234, M = 12.345 y M = 123.456.

Clave de 32 bits: p = 64.693, q = 60.811					
Clave pública e / Mensaje M	Vueltas necesarias para romper el secreto del mensaje				
	M = 12	M = 123	M = 1.234	M = 12.345	M = 123.456
e = 23	605.774	3.634.644	10.903.932	10.903.932	1.817.322
e = 421	13.169	39.507	118.521	118.521	39.507
e = 6.941	605.774	1.817.322	5.451.966	5.451.966	1.817.322
e = 37.867	302.887	3.634.644	10.903.932	10.903.932	1.817.322
e = 65.537	605.774	1.211.548	3.634.644	3.634.644	605.774

Figura 47. Vueltas necesarias en función de la clave pública y del valor de ataque.

Ataque por cifrado cíclico a claves con primos seguros

El uso de primos seguros incrementará la dificultad de realizar este ataque. Un ejemplo con tres claves muy parecidas de 16, 20, 24, 28 y 32 bits.

En cada una de las mediciones, en la clave intermedia K_2, K_5, K_8, K_{11} y K_{14} los valores de p y q serán primos seguros, y en las otras claves p y q serán valores primos no seguros inmediatamente inferiores a esos primos seguros en el primer caso e inmediatamente superiores en el segundo. Se usará la misma clave pública e y los mismos números de ataque 12.345 y 70.758 para las tres claves.

Se generan las claves, se ejecutan los ataques con los valores indicados y se obtienen las tablas que se recogen en la siguiente figura. Si se usan además distintos números como comienzo del ataque, en la mayoría de los casos se obtienen estos mismos valores. La siguiente figura muestra en bloques de 3 las claves generadas y atacadas de 16, 20, 24, 28 y 32 bits.

Clave	p	q	n	e	CPP	MNC	Vueltas
K_1	173	257	44.461	23	3	9	336
K_2	179	263	47.077	23	1	9	5.720
K_3	181	269	48.689	23	3	9	132

Clave	p	q	n	e	CPP	MNC	Vueltas
K_4	709	859	609.031	41	5	15	1.740
K_5	719	863	620.497	41	1	9	38.485
K_6	727	877	637.579	41	5	15	990

Clave	p	q	n	e	CPP	MNC	Vueltas
K_7	1.483	2.029	3.009.007	421	77	91	468
K_8	1.487	2.039	3.031.993	421	1	9	377.678
K_9	1.489	2.053	3.056.917	421	11	169	90

Clave	p	q	n	e	CPP	MNC	Vueltas
K_{10}	14.419	15.761	227.257.859	131	1	33	19.404
K_{11}	14.423	15.767	227.407.441	131	1	9	2.029.615
K_{12}	14.431	15.773	227.620.163	131	1	393	2.628

Clave	p	q	n	e	CPP	MNC	Vueltas
K_{13}	55.609	64.303	3.575.825.527	2.011	41	49	16.380 y 8.415
K_{14}	55.619	64.319	3.577.358.461	2.011	1	9	223.562.416
K_{15}	55.621	64.327	3.577.932.067	2.011	5	217	17.850 y 53.550

Figura 48. Ataques a claves de 16 a 32 bits con primos seguros y no seguros.

Se observa claramente que al usar primos seguros, además de generar una clave óptima, se dificulta de manera notoria este ataque.

4.1.3. Ataque por paradoja del cumpleaños

Aunque su nombre es el de paradoja, en realidad no es tal porque no hay nada en este problema que nos indique una contradicción matemática. Se trata, simplemente, de una serie cuyo resultado parece a simple vista asombroso y de ahí el nombre de paradoja.

El desarrollo del problema podría ser el siguiente: en un aula con decenas de personas, se marca en la pizarra mediante un recuadro todos los días del año, que se supondrá no bisiesto. Las personas abandonan el aula y entran nuevamente a ella de una manera aleatoria, de uno en uno, marcando con un aspa el recuadro en el que aparece el día de su cumpleaños en esa pizarra. El proceso termina cuando hay una colisión, es decir, cuando una persona que entra al aula ve que la fecha de su cumpleaños ya está marcada.

Se repite este proceso (limpiar la pizarra, todos fuera del aula y entrando de uno en uno) unas cuantas veces. La pregunta es, ¿cuántas personas deben entrar para que, en media, exista una probabilidad mayor que el 50% (lo que se conoce como confianza) de que ya esté marcado ese cumpleaños?

Aunque el resultado nos pueda sorprender, y de ahí su nombre de paradoja, en media basta que entren 23 personas para que la siguiente persona tenga ya más de un 50% de probabilidad de que su cumpleaños esté marcado. Y esto tiene sentido porque la primera persona que entra al aula se encuentra con los 365 días del año no marcados, la segunda se encuentra con un día menos libre de los 365, la tercera con dos días menos y así sucesivamente, por lo que va aumentando la probabilidad de que exista una colisión.

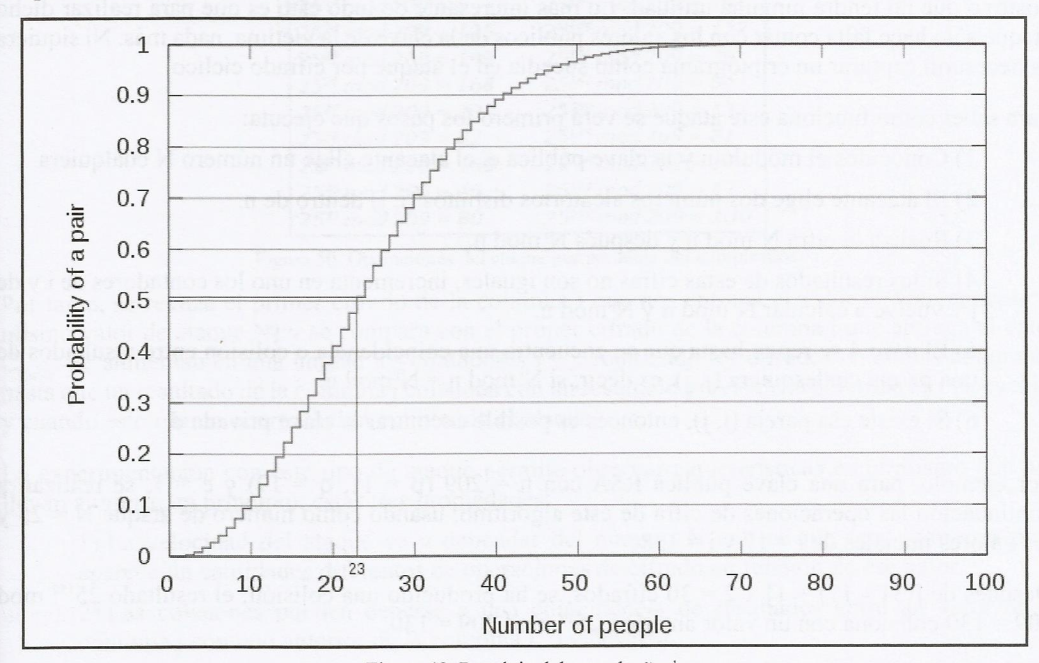


Figura 49. Paradoja del cumpleaños[1].

Explicación: si las personas entran en el aula de una en una y van marcando en esa pizarra con 365 días su cumpleaños, el primero tendrá una probabilidad de que su cumpleaños no esté borrado igual a n/n = 1, el segundo de (n-1)/n, el tercero (n-2)/n, etc. La probabilidad de no coincidencia será $p_{NC} = n!/n^k(n-k)!$ siendo k el número de personas.

Si k = 23, se tiene que p_{NC} = 0,493 y, por tanto, la probabilidad de coincidencia p_C o colisión será el complemento a la unidad, (1 - 0,493), es decir, p_C = 0,507 que es mayor que el 50%. Todo esto es muy interesante, pero ¿es de alguna de utilidad para realizar un criptoanálisis a RSA? Pues sí, y mucho como veremos más adelante. De hecho, el ataque basado en la paradoja del cumpleaños es el talón de Aquiles de las funciones hash que son necesarias entre otras cosas para permitir la firma digital.

En el año 1981 *Ralph Merkle* y *Martin Hellman* proponen un método para atacar al algoritmo DES, *Data Encryption Standard*, en su modalidad de doble cifrado con texto en claro y criptograma conocidos, basado en el problema o paradoja del cumpleaños y que deriva finalmente en lo que

1 *https://es.wikipedia.org/wiki/Paradoja_del_cumplea%C3%B1os*

conocemos como ataque por encuentro a medio camino, *meet in the middle*, y que finalmente hace desaconsejable este tipo de cifrado doble porque aumenta la fortaleza del sistema en sólo en un bit.

Este mismo principio nos permitirá encontrar colisiones en las cifras realizadas con RSA. Esta particularidad, conocida como ataque por paradoja de cumpleaños, cuando prospera permitiría encontrar la clave privada d, una clave privada pareja d' o bien, en muy pocos casos, un falso positivo que no tendrá ninguna utilidad. Lo más interesante de todo esto es que para realizar dicho ataque sólo hace falta contar con los valores públicos de la clave de la víctima, nada más. Ni siquiera es necesario capturar un criptograma como sucedía en el ataque por cifrado cíclico.

Para saber cómo funciona este ataque se verá primero los pasos que ejecuta:

1) Conocidos el módulo n y la clave pública e, el atacante elige un número N cualquiera.

2) El atacante elige dos números aleatorios distintos (i, j) dentro de n.

3) Realiza la cifra N^i mod n y después N^j mod n.

4) Si los resultados de estas cifras no son iguales, incrementa en uno los contadores de i y de j y vuelve a calcular N^i mod n y N^j mod n.

5) El paso 4 se repite hasta que se encuentra una coincidencia o colisión entre resultados de una pareja cualesquiera (i, j), es decir, si N^i mod n = N^j mod n.

6) Si existe esa pareja (i, j), entonces es posible encontrar la clave privada d.

Por ejemplo, para una clave pública RSA con n = 209 (p = 11, q = 19) y e = 7, se realizan a continuación las operaciones de cifra de este algoritmo, usando como número de ataque N = 25 y los valores iniciales de i = 17 y j = 138.

Después de [(31 - 17) + 1] x 2 = 30 cifrados, se ha producido una colisión: el resultado 25^{152} mod 209 = 130 colisiona con un valor anterior, 25^{17} mod 209 = 130.

En la práctica en lugar de elegir (i, j) de manera aleatoria como en el ejemplo anterior, es recomendable que se cubra todo el espacio del módulo. Así, se divide el espacio de n en dos mitades iguales, una zona izquierda o baja representada por el contador i de forma que $1 \leq i < n/2$, y una zona derecha o alta representada por el contador j de forma que $n/2 \leq j < n$, siendo lógicamente i y j enteros.

En la zona baja del módulo con el contador i se realizarán los siguientes cálculos:

$$N^1 \bmod n = C_{i1};\ N^2 \bmod n = C_{i2};\ N^3 \bmod n = C_{i3};\ \ldots\ N^{n/2-2} \bmod n = C_{i(n/2-2)};\ N^{n/2-1} \bmod n = C_{i(n/2-1)}$$

Y de la misma manera en la zona alta del módulo con el contador j se realizarán los siguientes cálculos:

$$N^{n/2} \bmod n = C_{j(n/2)};\ N^{n/2+1} \bmod n = C_{j(n/2+1)};\ N^{n/2+2} \bmod n = C_{j(n/2+2)};\ \ldots\ N^{n-2} \bmod n = C_{j(n-2)};$$

$$N^{n-1} \bmod n = C_{j(n-1)}$$

Esto es mejor verlo representado en dos columnas, donde el eje del tiempo será el desplazamiento vertical hacia abajo como se aprecia en la siguiente figura.

25^{17} mod 209 = **130**	25^{138} mod 209 = 159
25^{18} mod 209 = 115	25^{139} mod 209 = 4
25^{19} mod 209 = 158	25^{140} mod 209 = 100
25^{20} mod 209 = 188	25^{141} mod 209 = 201
25^{21} mod 209 = 102	25^{142} mod 209 = 9
25^{22} mod 209 = 42	25^{143} mod 209 = 16
25^{23} mod 209 = 5	25^{144} mod 209 = 191
25^{24} mod 209 = 125	25^{145} mod 209 = 177
25^{25} mod 209 = 199	25^{146} mod 209 = 36
25^{26} mod 209 = 168	25^{147} mod 209 = 64
25^{27} mod 209 = 20	25^{148} mod 209 = 137
25^{28} mod 209 = 82	25^{149} mod 209 = 81
25^{29} mod 209 = 169	25^{150} mod 209 = 144
25^{30} mod 209 = 45	25^{151} mod 209 = 47
25^{31} mod 209 = 80	25^{152} mod 209 = **130**

Figura 50. Operaciones del ataque por paradoja del cumpleaños.

Por tanto, se realiza el primer cifrado de la columna i que nos entrega el valor C_1 (obviamente el mismo valor de ataque N) y se compara con el primer cifrado de la columna j que entrega el valor $C_{j(n/2)}$. Se aumentan en una unidad los contadores i y j, repitiendo los cifrados en ambas columnas, hasta que un resultado de la columna j colisiona con un resultado previo de la columna i (o viceversa) y cuando esto último ocurre, habrá culminado el ataque.

La experimentación con este tipo de ataque permite observar características en el mismo que nos llevan a inferir, en principio, estas tres propiedades:

1) La velocidad del ataque va a depender del número N con el que éste comienza pues aparecerán cantidades diferentes de operaciones de cifrado en función de ese valor.

2) Las colisiones pueden deberse a una coincidencia de resultados entre un valor de la columna j con uno anterior de la columna i, o viceversa.

3) La tercera y muy importante: que la colisión se manifiesta por un último valor de la columna j que ha coincidido con el primer valor de la columna i, o viceversa, un último valor de la columna i ha coincidido con el primer valor de la columna j.

Lo interesante de esta última propiedad es que sólo hace falta conocer los dos primeros valores del ataque, esto es C_1 y $C_{n/2}$, de forma que en los siguientes cálculos de cifra de la columna de la derecha del contador j se buscará una colisión con C_1, y en los siguientes cálculos de cifra de la columna de la izquierda del contador i se buscará una colisión con $C_{n/2}$. Esto es así porque cuando se produce una colisión, los resultados de las cifras en las columnas i y j quedan en fase, encadenados, permaneciendo iguales de aquí en adelante, como se verá en el siguiente ejemplo.

Para la clave RSA con n = 133 y e = 5, se repiten las ecuaciones del algoritmo durante 12 cifrados en i y 12 en j para N = 25.

Como se observa en la columna de la derecha, la cifra 25^{73} mod 133 arroja el resultado 25 que corresponde al primer cifrado de la columna de la izquierda. Consecuentemente los resultados de los cifrados siguientes de i y de j serán iguales (93, 64, 4, 100, …).

Volviendo al algoritmo propuesto por *Merkle* y *Hellman*, una vez encontrada una colisión entre un resultado de la cifra con el contador i y un resultado de la cifra con el contador j, se deberán aplicar los pasos que se indican a continuación.

Los pasos del ataque de *Merkle* y *Hellman* en RSA son:

1) Se leen los valores del contador i y del contador j en los que se ha producido la colisión.
2) Como el atacante conoce la clave pública e, calcula:

$$w = (i - j) / \text{mcd}(e, |i - j|)$$

3) Deberán existir dos valores (s, t) de forma que se cumpla lo siguiente:

$$w*s + e*t = 1 \text{ (en mod e y en mod w)}$$

4) Las posibles soluciones a esta ecuación son:

$$w*s \bmod e = 1 \quad y \quad e*t \bmod w = 1$$

5) Se calcula s = inv (w, e)
6) Se calcula t = inv (e, w)
7) Se comprueba que w*s + e*t = 1
8) El valor t será la clave privada, una clave privada pareja o bien un falso positivo

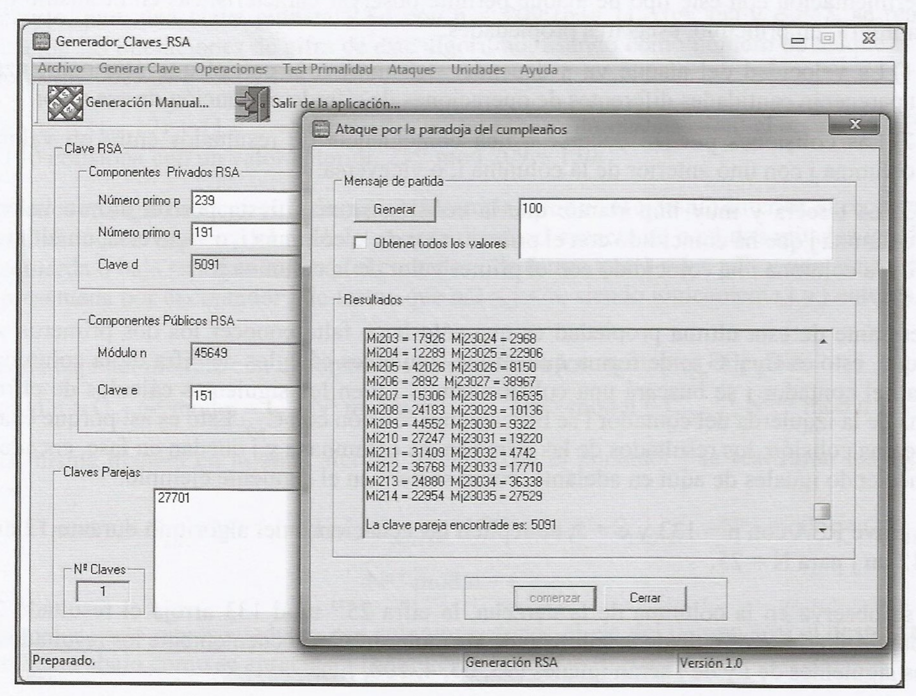


Figura 51. Ataque por paradoja del cumpleaños a una clave RSA.

```
CLAVE GENERADA

Número primo P generado:
239
Número primo Q generado:
191
Componente (módulo n) generado:
45649
Componente d privado generado:
5091
Componente e generado:
151

Paradoja del cumpleaños

Mi3 = 41371   Mj22824 = 22954
Mi4 = 28690   Mj22825 = 12950
Mi5 = 38762   Mj22826 = 16828
... ... ... ... ... ... ... ..
Mi212 = 36768   Mj23033 = 17710
Mi213 = 24880   Mj23034 = 36338
Mi214 = 22954   Mj23035 = 27529

La clave pareja encontrada es: 5091
```

Figura 52. Cifrados necesarios para encontrar la clave privada.

Se comprobará a continuación que tras seguir estos pasos, el ataque por paradoja del cumpleaños a una clave RSA encuentra la clave privada. Con el programa genRSA se genera una clave con p = 239, q = 191, e = 151 y se realiza un ataque por paradoja del cumpleaños siendo el valor de ataque N = 100. Las anteriores figuras muestran la clave generada y el resultado del log del ataque.

Como se observa, se ha encontrado la clave privada realizando solamente en 212*2 = 424 cifrados (el software genRSA comienza el contador i en 3) en un cuerpo de cifra cuyo tamaño es 45.649, cien veces mayor. Se comprobará el resultado aplicando las ecuaciones correspondientes.

Como e = 151 y ha encontrado colisiones en i = 214 y j = 22.824, se tiene:

w = (i - j)/mcd (e, |i - j|) = (214 - 22.824)/mcd (151, |22.610|) = - 22.610/1 = - 22.610

e * t mod w = 1 = 151 * t mod 22.610 = 1, luego t = inv (151, 22.610) = 5.091 (clave privada d)

Encontrará más información sobre este ataque y otros resultados del mismo en que se encuentra una clave privada pareja o un falso positivo en la Lección 10 del MOOC Crypt4you El algoritmo RSA (véase bibliografía).

Aunque este ataque presenta interesantes propiedades y características, entre ellas la posibilidad de realizar un ataque en red mediante divide y vencerás, un tema en el que los autores de este libro se encuentran trabajando a fecha de publicación de este libro, los altos valores de claves usados en la actualidad, sobre los mil bits y recomendable 2.048 bits, son aún un gran escollo para intentar vulnerar este algoritmo.

4.1.4. Recuperando textos en claro con exponente e pequeño

En el libro se ha justificado sobradamente porqué el uso del exponente pequeño e tiene una enorme utilidad (velocidad de cifrado, forzar el tamaño de la claves privada d al orden de n, etc.). Sin embargo, cualquier valor pequeño no es recomendable pues diferentes ataques han sido publicados para exponentes públicos bajos. En la actualidad, se recomienda el uso del número 4 de *Fermat* si bien nuevos ataques podría forzar a utilizar un número superior a éste, siempre pequeño en bits comparado con n.

Los ataques típicos documentados en la bibliografía (*Stereotyped Message Attack, Related Message Attack, Random Padding Attack*, etc.) muestran cómo el uso de exponentes *e* bajos pueden facilitar la recuperación del texto en claro. Esto se ha demostrado analíticamente en exponentes como puede ser e = 3.

Por ejemplo, el ataque "*related message*" permite recuperar el texto en claro de dos criptogramas muy parecidos siempre que se use un exponente bajo vulnerable. Imagine el caso que un usuario A envía un mensaje a B y por algún motivo tiene que retransmitirle de nuevo el mismo mensaje pero con un pequeño cambio. En este caso este tipo de ataque permite recuperar el mensaje en claro.

Otro tipo de ataque como el "*random padding*" destaca la necesidad de seguir una política adecuada de relleno de los mensajes a cifrar (el mensaje tiene que ser del tamaño de n para que no sea más fácil atacar por fuerza bruta al mensaje). Si el *padding* no tiene unas propiedades aleatorias adecuadas y se utilizan exponentes pequeños vulnerables, es posible recuperar el texto en claro.

4.2. Seguridad de la criptografía pública en el mundo real

En el apartado anterior se han analizado e implementado diferentes ataques matemáticos y computacionales al algoritmo RSA. Aunque los criptólogos deben estar pendientes de la publicación de nuevas técnicas de criptoanálisis y del aumento de la capacidad de computación de los atacantes, en la práctica la mayor parte de los problemas con tecnologías que utilizan criptografía pública, y en particular el algoritmo RSA, se debe a malos usos, fallos de implementación o usuarios poco formados en su uso correcto. En el mundo real la criptografía no se rompe, se esquiva.

A continuación se van a enumerar algunas de las vulnerabilidades más nombradas en tecnologías de amplio espectro que utilizan como base la criptografía de clave pública y el algoritmo RSA. En concreto las dos tecnologías que permiten hoy día seguridad en las comunicaciones mundiales a nivel civil: el protocolo SSL y las infraestructuras de clave pública (autoridades de certificación, certificados digitales, etc.). Comprender estos problemas facilitará la construcción de sistemas telemáticos más seguros y el uso correcto de la criptografía de clave pública en los mismos.

4.2.1. Problemas derivados de fallos de implementación

El papel lo aguanta todo, y eso lo saben bien los criptólogos. Un algoritmo puede ser inviable de invertir analíticamente e intratable de atacarlo computacionalmente pero su implementación puede abrir nuevas vías de ataque, tanto por software como por hardware, intencionadas o no. En los últimos años se han publicado una serie de vulnerabilidades de gran calado derivadas de fallos de implementación de algoritmos criptográficos. Uno de los más significativos fue la vulnerabilidad

de OpenSSL anunciada en mayo de 2008. El investigador argentino *Luciano Bello* descubrió que se habían implementado incorrectamente funciones aleatorias que se utilizaban en OpenSSL/Debian.

En concreto el fallo vino debido a la supresión de líneas de código en el fichero md_rand.c que afectó al generador de números pseudoaleatorios (PRNG). Por ejemplo, en plataformas Linux la generación de números aleatorios quedó reducida a la utilización del valor "aleatorio" ID (identificador del proceso) cuyo valor máximo es 32.768 (atacable por fuerza bruta). Esto producía material "aleatorio" predecible que facilitaba invertir los procesos criptográficos, y como consecuencia de ello certificados X.509, claves SSH e incluso material cifrado se vieron expuestos. Fue posible reconstruir claves privadas a través de claves públicas. Sin funciones criptográficas aleatorias los criptosistemas quedaron indefensos.

Si desea conocer los detalles es recomendable la lectura y visionado del trabajo *Predictable RNG in the Vulnerable Debian OpenSSL Package, the What and the How by Luciano Bello & Maximiliano Bertacchini* publicado, por ejemplo, en la conferencia Defcon16.

4.2.2. Autoridades de certificación y PKI. Falsificando certificados digitales

En apartados anteriores se destacó la importancia del uso de los certificados digitales y las infraestructuras de clave pública en las comunicaciones seguras en Internet. La criptografía pública sin ellos no es suficiente ya que es factible realizar ataques de hombre en el medio al no tener garantía de si la clave pública del destinatario es precisamente suya. Centrémonos en este apartado exclusivamente en los certificados digitales y en la posibilidad de realizar ataques basados en ellos.

Típicamente por su uso en Internet, se entiende que un certificado digital es un documento firmado electrónicamente por un prestador de servicios de certificación que vincula unos datos de verificación de firma a un firmante y confirma su identidad, siendo un documento que permite al firmante identificarse en Internet, tanto con las administraciones públicas como con numerosas entidades privadas. En resumidas cuentas, un usuario se fía de la entidad que certifica (firma) la validez de los certificados (clave pública) de otros usuarios.

Si se presupone que no es posible atacar al algoritmo criptográfico de clave pública, típicamente en Internet RSA, los vectores de ataque en principio más razonables podrían ir destinados a falsificar su contenido (su validez), bien atacando a la estructura del certificado digital (clave pública) o al procedimiento que emplean las entidades que los firman/generan y que certifican su validez.

Algunos ejemplos clásicos en la falsificación de certificados digitales no tiene que ver tanto con la manipulación de claves de criptografía pública de los certificados sino sobre todo ataques a algoritmos criptográficos débiles, históricamente algoritmos de hash, utilizados para la firma (certifica su validez). En otras ocasiones la falsificación de certificados se ha logrado engañando a la autoridad responsable de firmarlos y darle validez. Recordemos algunos ejemplos conocidos:

En la 25 edición de la *Chaos Communication Congress* celebrada en Berlín en diciembre de 2008, investigadores (*Alexander Sotirov* et al.) crearon un certificado SSL[2] "válido" aprovechándose de las peculiaridades de emisión de ciertas autoridades de certificación, de un ataque de colisión al

2 *http://www.win.tue.nl/hashclash/rogue-ca/*

algoritmo criptográfico MD5 y de una importante capacidad de cálculo basada en una centena de consolas PlayStation. Es verdad que MD5 ya no se usa en las Autoridades de Certificación para calcular el hash del certificado que va firmado con su clave privada, pero el estándar actual SHA-1 también está comenzando a tener unos problemas similares a los de su homólogo.

Otro tipo de ataque interesante de resaltar fue descubierto por el investigador *Moxie Marlinspike*. Al crear un certificado SSL y enviarlo a una Autoridad Certificadora para que lo firme, el campo al que se le suele prestar más atención es el CN (Common Name) que especifica el nombre del servidor, como puede ser *www.ejemplo.org*. *Moxie Marlinspike* descubrió que los estándares de certificado X.509 y SSL definen la cadena CN como una cadena PASCAL (se declara la longitud de la cadena en la posición 0 y se pone la cadena en el resto de posiciones). Curiosamente la mayoría de software de procesamiento de certificados está escrito en C. Dicho software suele manejar la cadena como una cadena C, poniendo un NULL (\0) al final de la cadena para indicar dónde termina. El problema llega cuando alguien obtiene un certificado de la forma *www.bancolegitimo.com\0www.atacante.org*. Cuando se procesa por un navegador, sólo se leerá la primera parte, *www.bancolegitimo.com*, permitiendo falsificar fácilmente al banco. La solución más fácil a este problema es que las entidades certificadoras rechacen todos los certificados que contuvieran el carácter NULL.

Todo esto está muy bien pero si los algoritmos criptográficos no tuvieran problemas y si no fuera posible falsificar certificados, porque ya se hubieran "corregido todos los fallos", los atacantes no tendrían opciones, ¿verdad? Pues, no. Una opción adicional, que debería ser todavía más complicada, consistiría en atacar a las propias autoridades que generan/firman los certificados y generan las claves, ya sea para robarlos o crearlos a medida. Esto no debería ser factible, pero por desgracia en la práctica no se toman las medidas de seguridad adecuadas y cada vez más esto es un problema serio de gran actualidad. Simplemente se van a recordar algunos casos significativos en los últimos años con matices diferentes: Comodo, Diginotar, virus Flame y certificado Adobe.

- El robo de certificados firmados por una autoridad de confianza puede suponer muchos problemas. Un ejemplo famoso en Marzo de 2011 fue la línea de negocio de certificados SSL de la empresa de seguridad Comodo. Uno de sus *partners* (que vendía certificados SSL) fue comprometido de forma que lanzó peticiones de firmado de certificados SSL sin la correspondiente verificación. Esto produjo la emisión de varios certificados digitales SSL falsos, es decir, certificados válidos para determinados sitios como *mail.google.com, www.google.com, login.yahoo.com, login.skype.com, addons.mozilla.org, login.live.com*, etc.

  Estos certificados eran completamente válidos para cualquier navegador Web. El impacto de este ataque implica que cualquier persona u organismo con capacidad de implementar un ataque de hombre en el medio, sería capaz de "transmitir" una web https falsa de Yahoo, Google y otros sin que el navegador protestase. Por suerte, estos certificados han podido revocarse por su número de serie; no obstante, este hecho destaca la problemática actual de Internet de obtener certificados digitales sin la adecuada validación por parte de los suministradores.

- Autoridad de certificación holandesa DigiNotar comprometida (julio 2011). Emisión fraudulenta de certificados para subdominios de *google.com* que podría permitir acciones como el descifrado de comunicaciones de otros usuarios con servicios variados (docs, mail, etc.). Es posible que la autoridad certificadora no sea muy conocida pero al estar permitida en la lista de autoridades de los navegadores web hacía que el potencial peligro fuera considerable.

Este es un claro ejemplo de lo que no se debe hacer en implementación y protección de una PKI. Al menos se deberían considerar las siguientes dos normativas que regulan cómo implementar una PKI y qué medidas de seguridad hay que aplicar: *Policy requirements for certification authorities issuing qualified certificates* (ETSI TS 101 456) y *Program for Certification Authorities* de Webtrust[3]. Por ejemplo, una CA raíz únicamente se emplea en dos ocasiones, o bien para crear CAs subordinadas o en caso de compromiso de esas CAs, para invalidarlas. Como caso real se puede recordar la CA raíz de la PKI del DNI electrónico español que se encuentra "protegida" en una celda con barrotes, totalmente offline (física y electrónicamente).

- Virus Flame y los certificados digitales (junio 2012). El investigador *Sergio de los Santos* de la empresa de seguridad española Hispasec resumió aquí sus detalles con claridad meridiana (véase bibliografía). El virus Flame es capaz de hacer un ataque MitM a los equipos de una red y suplantar *Windows Update* enviando una falsa actualización que infecte el equipo. Evidentemente para que un equipo Windows acepte una actualización desde *Windows Update*, ésta ha de estar firmada digitalmente por Microsoft. En este punto el virus Flame se aprovecha de la posibilidad de firmar código arbitrario mediante un certificado vinculado a licencias de Terminal Server. El problema viene al considerar que un certificado debe estar restringido en su uso a través de las extensiones *Extended Key Usage*. Así, un certificado puede ser usado para autenticar un servidor a través de SSL o para cifrar un correo (S/MIME). En la RFC 5280 donde se especifica la implementación, no se aclara totalmente el cómo hacerlo. Microsoft eligió la manera equivocada.

En resumen, el certificado final hereda EKU (*Extended Keys Usage*) "*por omisión*", debido a que las autoridades certificadoras superiores se lo proporcionan "*indirectamente*". Si se examina la cadena de certificación del certificado usado por TheFlame, se comprueba que finalmente, el certificado puede llegar a servir para firmar código, aunque no posea un campo EKU explícito. Con estos descubrimientos en la PKI de licencias de Terminal Server de Microsoft, los atacantes ya podían firmar código a partir de una licencia de Terminal Server. El código pasaría como totalmente válido (creado y validado por Microsoft) en varias versiones del sistema operativo (Windows 2003, XP, etc.). Podrían enviar una petición de firma a un servidor de licencias, y usar la clave privada de esa clave pública para firmar un programa y que pareciese totalmente de Microsoft. Esto es válido siempre que no se trate de Windows XP Service Pack 3, Vista, 2008 y 7

Estas versiones del sistema operativo en la validación de certificados tienen en cuenta una extensión que marca campos de certificados como críticos, de tal forma que si un certificado encuentra una extensión crítica que dice que debe ser usado para otro uso, el certificado se da por inválido. Los certificados de licencias de Terminal Server suelen tener la extensión "*Hydra*" marcada como crítica. Así que si los atacantes querían un certificado para firmar que funcionara en todas las versiones, tenían que eliminar por completo ese campo crítico. Esto lo consiguieron mediante colisiones al algoritmo MD5, el empleado en los certificados, de forma que fueron capaces de generar una licencia manipulada con firma de Microsoft y abrieron de esta forma la infección masiva de equipos.

- Certificado Adobe (septiembre 2012). La compañía Adobe a través de un post publicado en septiembre de 2012 "*Inappropriate Use of Adobe Code Signing Certificate*"[4] alertó de que,

3 http://www.webtrust.org/item27804.pdf

4 *https://blogs.adobe.com/asset/2012/09/inappropriate-use-of-adobe-code-signing-certificate.html*

al menos, un certificado digital para firmar código había sido comprometido y era necesario actualizar algunos de sus productos ante una inminente revocación del certificado digital con el que se encontraban firmados. Lo interesante del certificado comprometido es que fue utilizado para firmar la popular herramienta *PwDump7* - junto con la librería *libeay32.dll* de *OpenSSL* - que se utiliza para volcar las contraseñas de sistemas *Microsoft Windows* desde el *LSA*, y *myGeeksmail.dll* que es un filtro *ISAPI* que puede instalarse en servidores web para interceptar y manipular las conexiones *HTTP*, lo que permitiría a un atacante acceder a toda la información transmitida por ese servidor web. El firmado de herramientas de hacking con un certificado válido de compañías de renombre es casi un salvoconducto contra cualquier antimalware o similar, ya que estos cuentan con medidas especiales para no detectar/analizar como malware software firmado por entidades certificadoras de confianza.

En último lugar, es interesante recordar que existen otros tipos de ataques que tienen que ver más con lo confiable que sea la autoridad certificadora. Existen ataques que consistirían en que la CA o entidades comerciales generarán certificados válidos de ellas mismas para fuerzas gubernamentales o gobiernos (¿censores?).

Por ejemplo, en el mejor de los casos por orden judicial, permitiría hacer un ataque de hombre en el medio, difícilmente detectable, en tanto en cuanto el certificado sería válido y firmado por una autoridad competente. Esto abre una cuestión interesante de debatir ¿en cuántas autoridades de confianza debemos confiar? En el blog de seguridad informática SecurityByDefault el investigador *Yago Jesús*[5] resumió algunos de los datos significativos de las autoridades de certificación en las que "confía" por defecto un navegador web:

- 130 entidades de 45 países
- Estados Unidos lidera claramente el número de organizaciones acreditadas (26)
- España es el segundo país con más CAs acreditadas por Microsoft (11)
- Túnez es el único país del Magreb que tiene presencia acreditada (Turquía es el otro país musulmán acreditado)
- China tiene 2 organizaciones capaces de emitir certificados SSL y dos más adscritas a Hong Kong y Macao (provincias con régimen especial)
- Hay dos entidades acreditadas con sede en los paraísos fiscales de Bermudas y Singapur
- Existen múltiples CAs acreditadas que están ligadas a entidades gubernamentales

Estas organizaciones tienen potestad para certificar que un dominio, por ejemplo *www.google.com*, es quien dice ser. Un compromiso en cualquiera de ellas facilitaría operaciones fraudulentas o la monitorización, gubernamental o no, de ciudadanos. Esto es un tema importante especialmente si se piensa en la falta de transparencia de algunos países o la facilidad de corromper empresas con dinero en ciertos lugares. Si desea minimizar este problema, existen herramientas para limitar el número de autoridades de certificación en las cuales confía. Por ejemplo, la herramienta SSLCop del investigador *Yago Jesús* permite bloquear CAs por su procedencia geográfica[6].

5 *http://www.securitybydefault.com*

6 *http://www.securitybydefault.com/2012/03/sslcop-10.html*

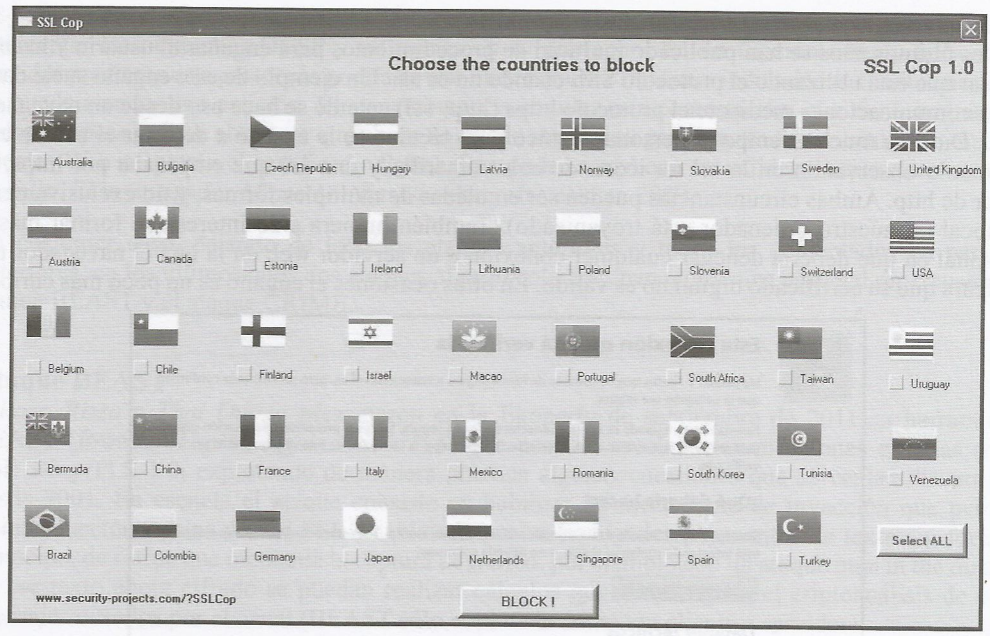


Figura 53. Herramienta SSLCop.

4.2.3. Seguridad del protocolo SSL. Engañando al usuario

El protocolo SSL/TLS es la referencia a la hora de hablar de seguridad en Internet (véase por ejemplo https). Este protocolo proporciona autenticación y privacidad de la información entre extremos sobre Internet mediante el uso de criptografía y una serie de fases básicas:

1) Negociar entre las partes el algoritmo que se usará en la comunicación

2) Intercambio de claves públicas y autenticación basada en certificados digitales

3) Cifrado del tráfico basado en cifrado simétrico

Dado su uso masivo en Internet, cualquier ataque sobre el mismo suele tener una gran repercusión. En los últimos años se han documentado multitud de ellos (algunos ya se han visto en apartados anteriores). Si desea una rápida introducción al tema le recomendamos el resumen *A brief chronology of SSL/TLS attacks*[7] y el artículo *Lessons Learned from previous SSL/TLS attacks. A brief Chronology of attacks and weaknesses*[8] de *Christopher Meyer* y *Jörg Schwenk*.

En cualquier caso, en la práctica la manera más fácil de vulnerar la seguridad proporcionada por el protocolo SSL/TLS consiste en engañar al usuario haciéndole pensar que lo está utilizando cuando en realidad no es así o hacerle pensar que está utilizando certificados válidos que autentican a las partes de la comunicación cuando tampoco es así. Esto se puede aprender fácilmente recordando algunos ataques documentados.

7 *http://armoredbarista.blogspot.de/2013/01/a-brief-chronology-of-ssltls-attacks.html*
8 *https://eprint.iacr.org/2013/049.pdf*

La 's' de seguro (https) y el candado amarillo

En los últimos años se han publicado multitud de procedimientos para engañar al usuario y hacerle pensar que está utilizando el protocolo SSL cuando no es así. Un ejemplo de este engaño suele darse en la comunicaciones mediante el protocolo https (http+ssl) cuando se hace uso desde un navegador web. Durante mucho tiempo al personal técnico y no técnico se le enseñó a detectar el uso de este protocolo observando en la url un icono de color amarillo y una url que empezaba por https, en lugar de http. Ambas circunstancias pueden ser emuladas de múltiples formas, y no exclusivamente en local (si nuestro ordenador está troyanizado). También hubiera sido interesante formar más al personal en que debería denegar cualquier conexión a un servidor web en la que el navegador nos indicara que su certificado digital no es válido. En otras ocasiones el engaño es un poco más curioso.

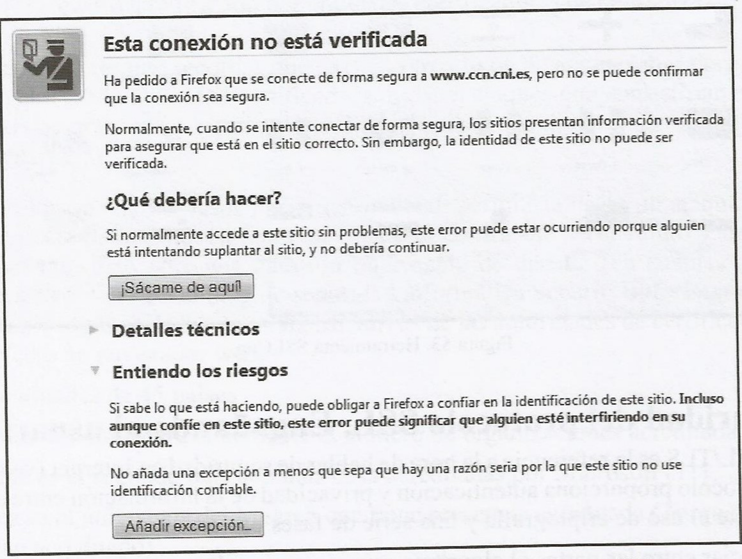


Figura 54. Ejemplo de aviso por navegador de certificado no válido.

En 2009, *Moxie Marlinspike* presentó en la conferencia de seguridad informática BlackHat la herramienta SSLtrip que automatiza un ataque de hombre en el medio al protocolo SSL. La idea es sencilla: cuando se llama a una página Web, se sustituyen todos los enlaces https por http, con la intención que la comunicación entre el cliente y el atacante sea por http y la comunicación entre atacante y servidor por https. Para engañar más al usuario, se aprovecha de ciertos trucos, como por ejemplo simular el "candado amarillo" cargando esta imagen en el *favicon*.

Clientes mal configurados y certificados fraudulentos

Cuando se detectan certificados digitales fraudulentos, suelen revocarse gracias al número de serie que incluyen y para esto suele utilizarse el protocolo OCSP (*Online Certificate Status Protocol*). Una configuración incorrecta de este protocolo simplifica los ataques al protocolo SSL.

OCSP es un protocolo de consulta online para saber si un determinado certificado digital ha sido revocado o no. Para ello, el cliente envía la petición a la dirección de la CRL (*Certificate Revocation List*), que viene indicada en el propio certificado digital. Si un atacante está haciendo un ataque

de hombre en el medio para utilizar uno de estos certificados digitales, entonces también puede interceptar las peticiones OCSP y utilizarlas en su provecho. En un funcionamiento normal, un servidor mediante este protocolo podría enviar una respuesta *Try Later* indicando al cliente que ahora no puede atender una petición. El atacante podría simular esta contestación, que tiene asignado el código 3, para indicar al cliente que ahora no puede atender su petición. Ante esta situación muchos clientes Web aceptarán el certificado digital al no poder corroborar su validez, lo que claramente es un fallo. El investigador *Moxie Marlinspike* demostró esto en 2009.

En cualquier caso, dejando de lado los ataques anteriores y todos los comentados en párrafos anteriores, en los últimos años dos ataques al protocolo SSL han tenido una especial repercusión: el ataque BEAST y el ataque CRIME.

Ataque BEAST

Juliano Rizzo y *Thai Duong* presentaron en la Ekoparty de septiembre de 2011 su herramienta BEAST (*Browser Exploit Against SSL/TLS*) que permite revelar comunicaciones cifradas sobre SSL 3.0 y TLS 1.0, explotando de manera práctica algunas cuestiones que se venían discutiendo desde 2001. En esencia el ataque consiste en habilitar un mecanismo de inyección que permita la introducción en una sesión SSL establecida por un navegador fragmentos de texto sin formato conocido, de forma que mediante la captura de tráfico, por ejemplo con un ataque man in the middle, de ese texto ahora cifrado se puedan realizar cálculos que simplifiquen el criptoanálisis de otros mensajes cifrados por el canal (BEAST sólo explota la conexión cliente a servidor).

El kit de herramientas BEAST utiliza esta capacidad para extraer cookies de sesión que se pueden usar para secuestrar la sesión de un usuario. La descripción del ataque con todo tipo de detalles puede ser accedido desde la bibliografía. Aunque una solución global al problema residiría en la actualización del protocolo a una versión más moderna, por ejemplo TLS 1.1, esto no siempre es factible. En la práctica, grandes compañías como Mozilla, Google o Microsoft actualizaron la forma en la que operar con sus productos para mitigar ataques tipo BEAST, especialmente cambiando la forma en la que se transmitían los paquetes cifrados o incluso alguno recomendando temporalmente no utilizar cifradores en modo CBC (AES o DES) y sí cifradores de flujo tipo RC4, o utilizar algoritmos de compresión para dificultar el análisis texto en claro inyectado con texto cifrado.

Adicionalmente a esto, cualquier recomendación que pudiera limitar el tiempo de vida y reutilización (asociación a la IP del cliente o similar) de una cookie/token de sesión de un usuario dificultaría su ataque (el atacante necesita tiempo para saber cómo descifrar la información cifrada). Esto parece trivial pero no debe olvidarse. No obstante, el ataque BEAST no fue el único ataque de este tipo al protocolo SSL y por tanto las recomendaciones a considerar son muchas más. Es destacable otro trabajo de los mismos autores publicado en la conferencia Ekoparty de 2012 en este sentido. En esta conferencia publicaron el ataque CRIME (*Compression Ratio Info-Leak Mass Explotaition*).

Ataque CRIME

Este ataque permitía mediante la inyección de código Javascript en el navegador de la víctima y el análisis de información cifrada previamente comprimida (longitud de los datos comprimidos) recuperar información, en principio de pequeño tamaño, intercambiada por el canal cifrado, típicamente credenciales de usuario (cookies de sesión). La idea del ataque es sencilla: si es posible inyectar código, podríamos predecir información presente en los datos cifrados teniendo en cuenta

el funcionamiento de los algoritmos de compresión. Esto es factible construyendo mensajes byte y byte y analizando si el número de bytes enviado es menor o no (mejor comprensión). Si es menor, el mensaje inyectado coincidirá con otro existente. De esta forma y mediante múltiples peticiones es posible descubrir, por ejemplo, información de una cookie de sesión intercambiada a través de SSL.

Curiosamente la utilización del algoritmo RC4 en lugar de cifrados en bloques como CBC o habilitar la compresión de datos hace factible el ataque (recuérdese que esas fueron unas de las soluciones temporales recomendadas contra BEAST). En cualquier caso, para que el ataque sea efectivo se necesita que la compresión se utilice en ambos extremos de la comunicación, que sea factible inyectar código en la máquina y poder acceder al tráfico intercambiado entre cliente y servidor. Todos los detalles del ataque y los escenarios donde es posible pueden consultarse en la bibliografía.

Como puede observar el lector, los protocolos criptográficos basados en clave pública unido a los certificados digitales y las PKI están siendo sometidos al escrutinio de todo tipo de investigadores publicando ataques criptoanalíticos, computacionales, basados en fallos de programación o configuración, etc. La tendencia actual de mitigación consiste simplemente en corregir posibles fallos y recomendar algoritmos más seguros. Por desgracia, no existen propuestas conceptuales interesantes que aborden el problema de otra forma. Algunos investigadores como *Moxie Marlinspike* han intentado aproximarse a esta cuestión, de momento con ideas muy matizables. Su propuesta, recordando casi el concepto de anillo de claves de confianza al estilo GPG, indica que el usuario sea el que decida si un certificado SSL es válido o no, a la vez que puede fiarse de un "notario" (persona/organización) en el que delegar la confianza, por lo que todos los certificados que él dé por buenos serán buenos para ti. Si desea probar esta propuesta existe una extensión para Firefox denominada Convergence[9].

En cualquier caso, el futuro está abierto en esta cuestión. Nuevas mentes allanarán el camino hacia nuevas soluciones.

4.2.4. Mitigaciones y recomendaciones para el uso de HTTPS

Antes de finalizar con este capítulo, se ve necesario realizar una serie de recomendaciones básicas para la mitigación de problemas en SSL/TLS. Las más básicas consistirían en utilizar la versión más moderna del protocolo SSL debidamente configurado e intentar que las implementaciones de dicho protocolo estén corregidas de fallos conocidos. En el mundo real esto no es siempre posible y depende de la necesidad de mantener la compatibilidad entre sistemas. Independientemente de esto, en el caso de su uso en la web, es necesario recodar algunos consejos mínimos para mitigar problemas.

1) Mantener el navegador web actualizado para que la implementación del protocolo SSL/TLS esté libre de vulnerabilidades conocidas y mitigue de la mejor forma los ataques conocidos. Por ejemplo, ataques de renegociación SSL[10].

2) Añadir en la url (cuando sea posible) la dirección directa https de la página a la que nos deseamos conectar. El add-on para Firefox "HTTPS Everywhere[11]" puede ayudar automatizar esto.

9 *http://convergence.io/*

10 *http://packetstormsecurity.com/0911-advisories/Renegotiating_TLS.pdf*

11 *https://www.eff.org/https-everywhere*

3) Denegar acceso a una página Web cuando el certificado no sea válido. Esto es especialmente crítico para el acceso a cuentas bancarias, datos personales, etc. En otro caso, el usuario baremará la relación riesgo-acceso a esa web.

4) Configurar los navegadores web para que hagan comprobaciones OCSP por defecto y si la conexión OCSP falla, el certificado por defecto no sea dado por bueno. Esto evitará ataques basados en negación de servicio al OCSP y uso de certificados revocados (menú Herramientas → Avanzado → Cifrado → Validación).

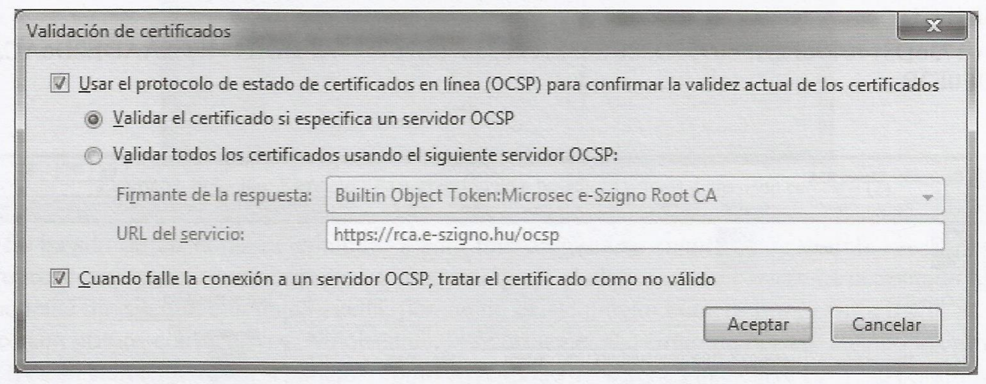


Figura 55. Configuración de OCSP en Firefox.

1) Ciertos complementos software podrían ayudar a detectar plagios. Por ejemplo, en Firefox el add-on *Certificate Patrol*[12] monitoriza cambios en servidores https de forma que si un día el certificado de SSL de Gmail es diferente al registrado, notifica ese cambio.

2) Limite el número de autoridades de confianza en las que confía. Para ello puede usar software como SSLCop[13].

3) Adicionalmente a estas protecciones mínimas puede analizarse la seguridad que presenta el servidor Web al que nos conectemos frente a diferentes ataques y cuestiones relacionadas con la seguridad del protocolo SSL. Un ejemplo de herramienta para hacer esto es TLSSLed[14] del grupo Taddong.

4.3. Ejercicios y prácticas

Ejercicio 1. Factorización de n basada en primos demasiado cercanos

En RSA se recomienda que los primos p y q estén separados algunos bits. ¿Qué sucedería si se eligen primos p y q muy cercanos? La respuesta es que en este caso el algoritmo de *Fermat* sería muy eficiente porque resuelve la factorización en muy pocos pasos. Por ejemplo, una clave RSA cuyo módulo n fuera 90 dígitos (297 bits) sería factorizada en unas 20 horas por el programa factor.exe

12 *https://addons.mozilla.org/en-us/Firefox/addon/certificate-patrol/*

13 *http://www.security-projects.com/?SSLCop*

14 *http://blog.taddong.com/2013/02/tlssled-v13.html*

como ya se ha visto mientras que utilizando el algoritmo de *Fermat* la solución sería casi inmediata como muestra el ejemplo.

Se genera manualmente una clave con genRSA, utilizando e = 131 (0x83) y p, q con los siguientes valores:

p = 166598033b71b5c684031b82966169ed2d5651

q = 166598033b71b5c684031b82966169ed2d5671

n = 01F59E71DB08499907A84EE46ACD68BEE564D41BB51023D09AB619B3ECE2840F5789153F4FC1

d = 01736D8140796F2AF60953D22BE838104825876E767517AFDDB444FC5AADE2D6ACA575DE0F2B

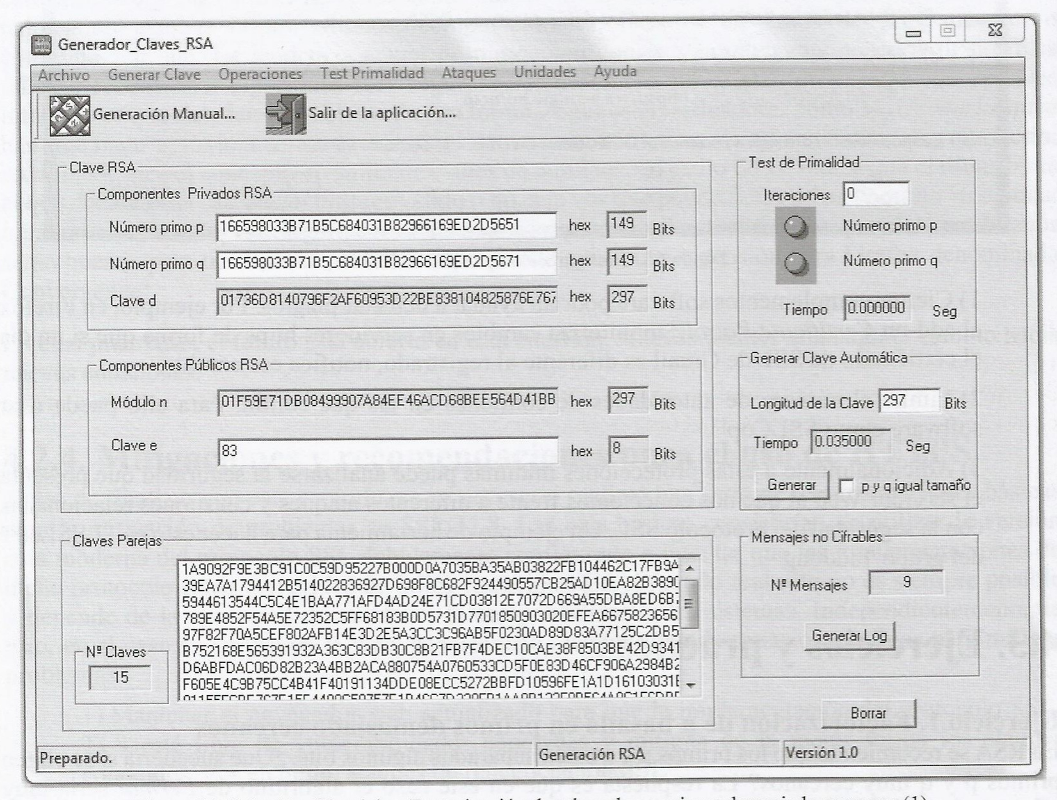


Figura 56. Solución: Ejercicio - Factorización de n basada en primos demasiado cercanos (1).

Con el mismo programa se elige Ataque por factorización y se indica que ejecute solamente 1 vuelta, obteniendo de forma inmediata lo siguiente:

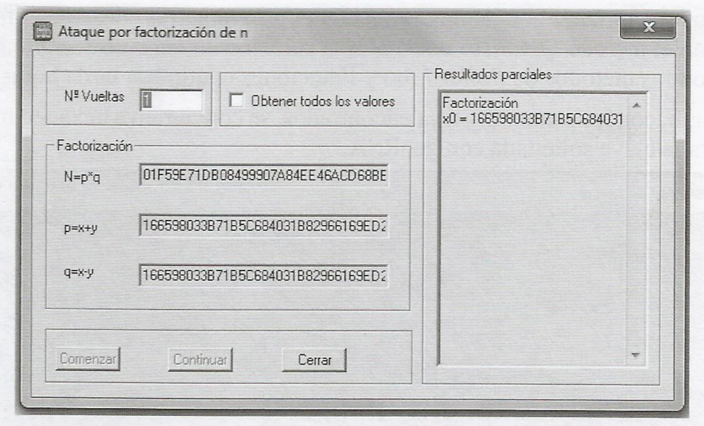


Figura 57. Solución: Ejercicio - Factorización de n basada en primos demasiado cercanos (2).

Se ha bajado de 20 horas de cómputo a sólo microsegundos, simplemente usando un algoritmo apropiado para primos muy cercanos. Como es lógico, el algoritmo de *Fermat* ha necesitado hacer solamente un cálculo, una única vuelta, porque no existe ningún número primo entre p y q, ya que éstos son primos consecutivos. No obstante, si se utiliza el algoritmo de *Fermat* cuando los primos p y q están suficientemente separados, como en realidad debe ser en una clave RSA, su rendimiento es muy pobre y es mejor usar otros algoritmos tal y como lo hace el programa factor.exe.

Ejercicio 2. Factorización mediante *Pollard* Rho, Dixon y Fracciones continúas

Usando primero el programa factor.exe y luego Fortaleza de Cifrados, ataque por factorización del módulo esta clave RSA con primos de 100 bits cada uno y relativamente cercanos, aunque no correlativos como en el ejemplo visto anteriormente:

```
n = 437610495647369551336533004629577846576256680060159623373471
e = 65537
```

Para el caso del software Fortaleza de Cifrados, haga la operación con los tres algoritmos implementados: *Pollard* rho, Dixon y Fracciones continuas. Una vez encontrados p y q, convierta estos valores en hexadecimales con el software Dec2Hex y genere con genRSA esa clave RSA siendo e = 0x010001. Hecho esto, use genRSA para realizar un ataque por factorización que usa el algoritmo de *Fermat*. Observe que también tarda muy poco en encontrar la solución.

¿Qué ha sucedido con los tiempos de cómputo al factorizar n con los programas genRSA, factor y Fortaleza de Cifrados? ¿Qué diferencias observa entre los primos p y q encontrados?

Solución:

1) Para realizar la factorización del módulo n con el programa factor, se usa el programa por lotes timer.bat que se encuentra en el trabajo *Ejercicio práctico del problema de la factorización entera: entendiendo la fortaleza de RSA*[15], con el comando *timer factor NUM*, obteniendo un tiempo de 37 segundos como se observa en la siguiente figura.

15 *http://www.criptored.upm.es/guiateoria/gt_m001c1.htm*

2) Se intenta factorizar ahora el módulo n con el programa Fortaleza de Cifrados.
   2.1. Con el método Dixon, se encuentra la solución en tan sólo 3 segundos.
   2.2. Con los métodos de *Pollard* rho y Fracciones continúas, tarda horas... (no medido).

3) Por último, con los primos p y q ya encontrados, se convierten a hexadecimal con Dec2Hex y se genera la clave solicitada con genRSA.

```
Símbolo del sistema
C:\Criptolab\EjercicioFreewareFactor>timer factor 437610495647369551336533004629
9623373471
first trying brute force division by small primes
now trying 1000 iterations of brent's method
now trying william's (p+1) method
phase 1 - trying all primes less than 10000
phase 2 - trying last prime less than 1000000
now trying pollard's (p-1) method
phase 1 - trying all primes less than 100000
phase 2 - trying last prime less than 5000000
now trying lenstra's method using 10 curves
curve   1 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
curve   2 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
curve   3 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
curve   4 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
curve   5 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
curve   6 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
curve   7 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
curve   8 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
curve   9 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
curve  10 phase 1 - trying all primes less than 20000
          phase 2 - trying last prime less than 2000000
finally - the multiple polynomial quadratic sieve - with large prime (*)
using multiplier k= 1 and 3164 small primes as factor base
working...  3114
trying...
working...* 3114
trying...
working...  3114
trying...
working...* 3114
trying...
PRIME FACTOR     661521349351105000008725817463
PRIME FACTOR     661521349351199998787654568217
Started at 14:17:11,08
Ran for 36880 ms

C:\Criptolab\EjercicioFreewareFactor>_
```

Figura 58. Solución: Ejercicio - Factorización de n en 37 segundos mediante factor.exe.

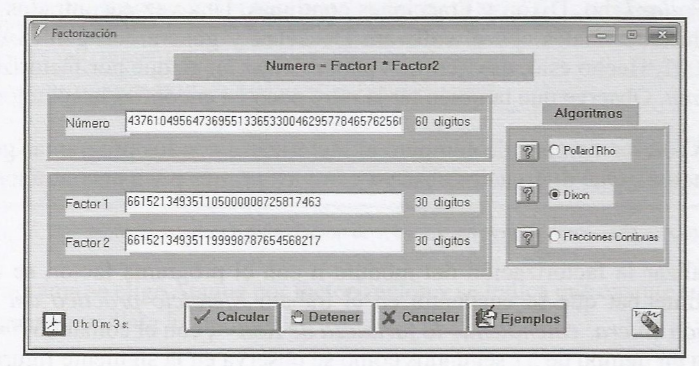


Figura 59. Solución: Ejercicio - Factorización de n mediante el programa Fortaleza y método Dixon que requiere 3 segundos.

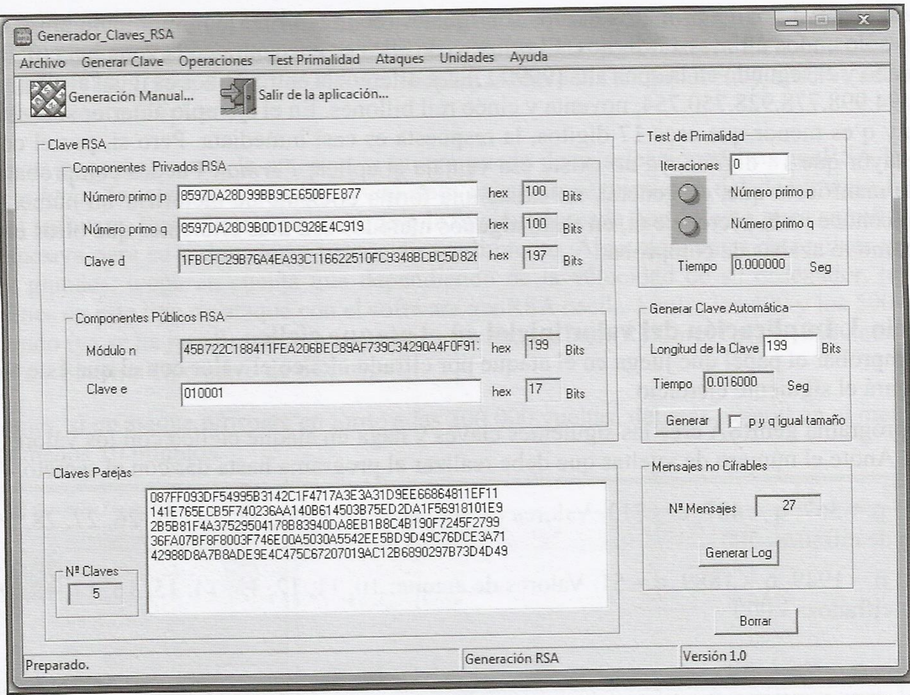


Figura 60. Solución: Ejercicio - Clave RSA generada con los dos primos de 100 bits encontrados y e=0x010001.

Se ejecuta luego el ataque por factorización, indicando que ejecute 2.000 vueltas, y tarda también 3 segundos en encontrar la solución tras 1.705 operaciones.

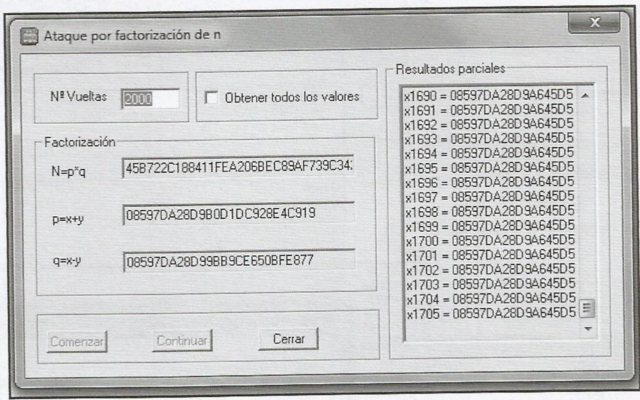


Figura 61. Solución: Ejercicio - Factorización de n mediante el programa genRSA y método *Fermat*. Módulo factorizado en vuelta 1706.

Se han obtenido dos primos de 30 dígitos:

p = 661521349351105000008725817463

q = 661521349351199998787654568217

Como se observa, los 13 primeros dígitos son iguales (6615213493511) y por tanto ambos primos difieren sólo en los últimos 17 dígitos. El primero de ellos se encuentra en la zona baja (0500 ...) de ese tamaño y el segundo en la zona alta (9999 ...). La diferencia entre ambos es igual al número de 17 dígitos 94.998.778.928.750.754, noventa y cinco mil billones. En el ejemplo anterior si la diferencia entre p y q es menor que esos 17 dígitos, la respuesta es casi inmediata. Pero si, por el contrario, fuese mayor que 18 dígitos, ya no existe esa ventaja al aplicar *Fermat*. Puedes comprobarlo. Esto pone de manifiesto que, en general, no existe una forma eficiente de factorizar un número entero si no se conoce nada acerca de él, en tanto algunos métodos son más eficientes que otros en ciertos casos como lo acabas de comprobar.

Ejercicio 3. Implicación del valor inicial en el ataque cíclico

Para comprobar el papel que juega en el ataque por cifrado cíclico el valor con el que éste se inicia, se realizará el siguiente ejercicio.

Con el programa genRSA cree las siguientes claves y haga un ataque cíclico con los valores que se indican. Anote el número de vueltas que debe realizar el programa hasta dar con el secreto.

Clave 1) p = 367, q = 487, e = 713. Valores de ataque: 20, 21, 22, 23, 24, 25, 26, 27, 28, 29 y 30. Número de cifrados: 500.

Clave 2) p = 1949, q = 1889, e = 51. Valores de ataque: 10, 11, 12, 13, 14, 15, 16, 17, 18, 19 y 20. Número cifrados: 3.000.

Solución:

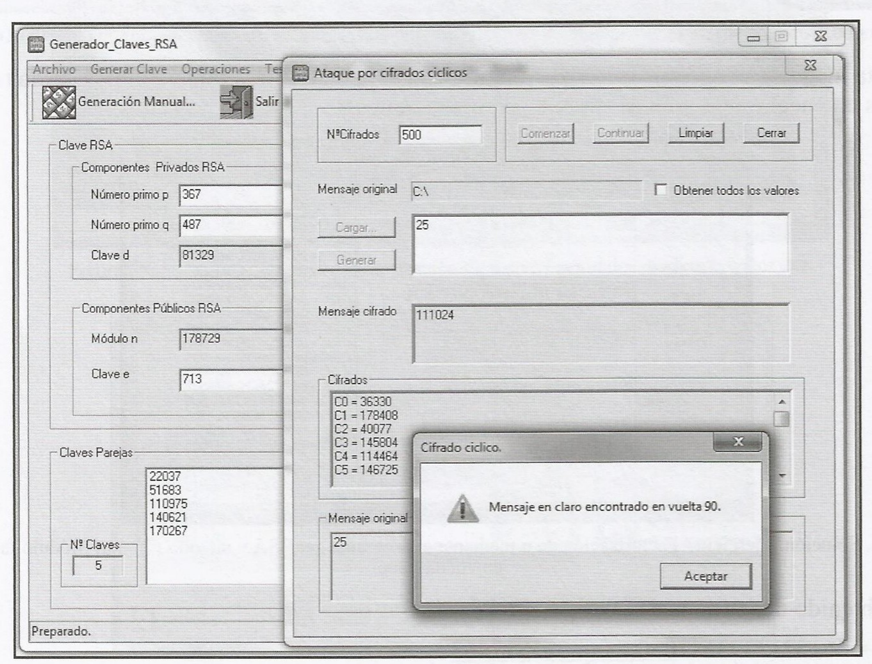


Figura 62. Solución: Ejercicio - Ataque por cifrado cíclico a la clave con n=178.729 y e=713.

Como ha podido comprobar, en función del valor de la entrada, se rompe el secreto para valores distintos de vueltas. En este caso y dependiendo de la entrada, para la clave 1 encontrará 90, 270 y 810, vueltas y 783, 1.566, 3.132 y 6.264 vueltas para la clave 2. Compruébelo y observe además que estos valores son múltiplos.

Para claves pequeñas como la anterior, si quiere puede dejar seleccionada en genRSA la opción *"Obtener todos los valores"* que le servirá para hacer un seguimiento del documento html que genera el software. Para claves mayores, deje la opción marcada por defecto pues significaría una carga excesiva para su ordenador y bajaría su rendimiento. Algo más; antes de realizar un ataque a claves grandes, tenga en cuenta que dependiendo de la velocidad de su ordenador, la tasa de operaciones en este tipo de ataque con el software genRSA oscilará entre los 400 y los 500 cifrados por segundo (ya se ha justificado esta tasa) en una máquina Intel(R) Core(TM) i7-2600 con CPU de 3,40 GHz y Sistema Operativo de 64 bits.

Por lo tanto, si un ataque prospera en torno a las 300.000 vueltas, deberás esperar en el mejor de los casos al menos 10 minutos.

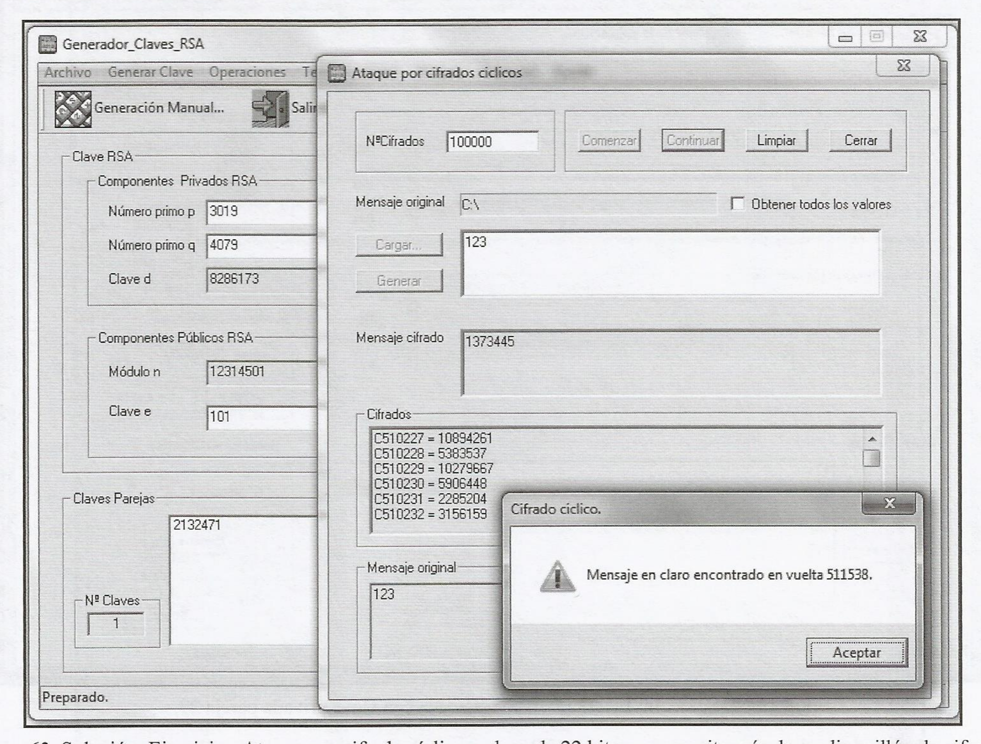


Figura 63. Solución: Ejercicio - Ataque por cifrado cíclico a clave de 22 bits que necesita más de medio millón de cifrados.

Ejercicio 4. Descubriendo la clave privada mediante ataque de paradoja de cumpleaños

Para n = 133 y e = 5, calcule la clave encontrada para la colisión encontrada entre 25^{1} mod 133 = 25 y 25^{73} mod 133 = 25.

Solución:

1) i = 1, j = 73
2) w = (i - j) / mcd (e, |i - j|) = (1 - 73) / mcd (5, |1 - 73|) = -72 / 1 = -72
3) Como w*s + e*t = 1, entonces -72*s + 5*t = 1
4) Posibles soluciones: w*s mod e = -72*s mod 5 =1 y e*t mod w = 5*t mod 72 = 1
5) Calcule s = inv (w, e) = inv (72, 5) = inv (3, 5) = 2 (porque 72 mod 5 = 3)
6) Calcule t = inv (e, w) = inv (5, 72) = **29**
7) Compruebe: w*s + e*t = -72*2 + 5*29 = -144 + 145 = 1
8) El valor t = 29 es una de las claves privadas parejas como se muestra en la siguiente figura.

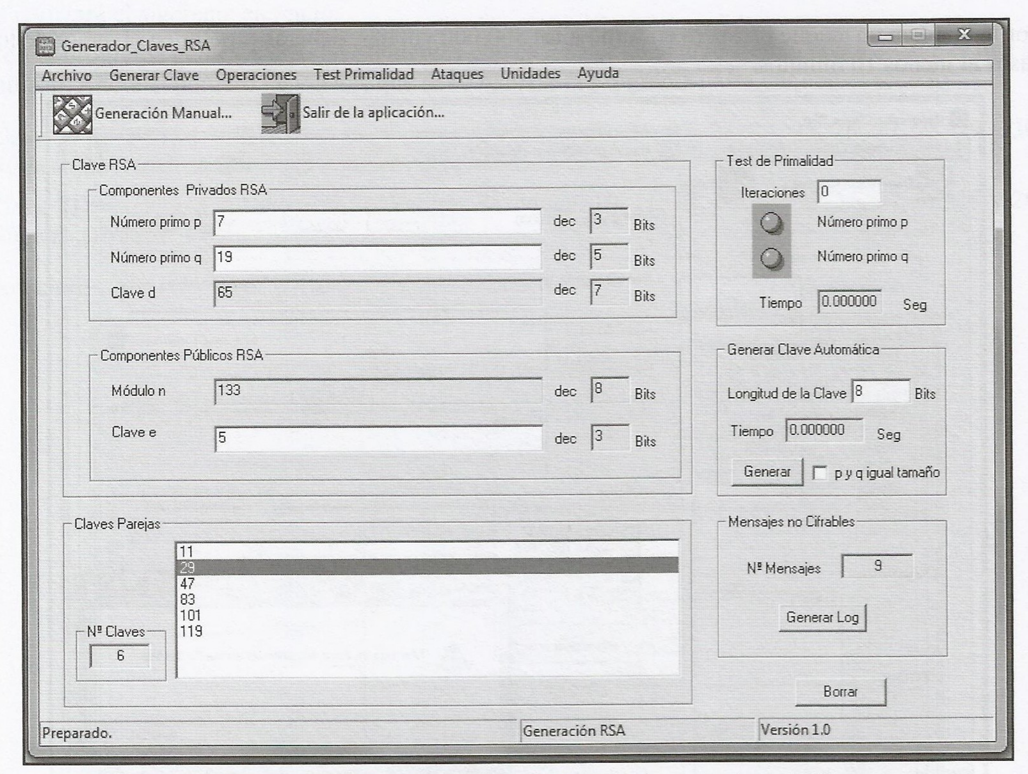


Figura 64. Solución: Ejercicio - Clave atacada con clave privada pareja 29.

Genere ahora dos claves en las que p = 239 y q = 191. La primera de ellas tendrá como clave pública e = 151 y la segunda e = 149. Realice el ataque con N = 100. Compruebe con el software que el ataque encuentra en el primer caso la clave privada y en el segundo una clave privada pareja. Es decir un pequeño cambio en el valor de la clave pública e produce este efecto. Demuestre a continuación estos valores usando las ecuaciones del algoritmo de ataque del cumpleaños.

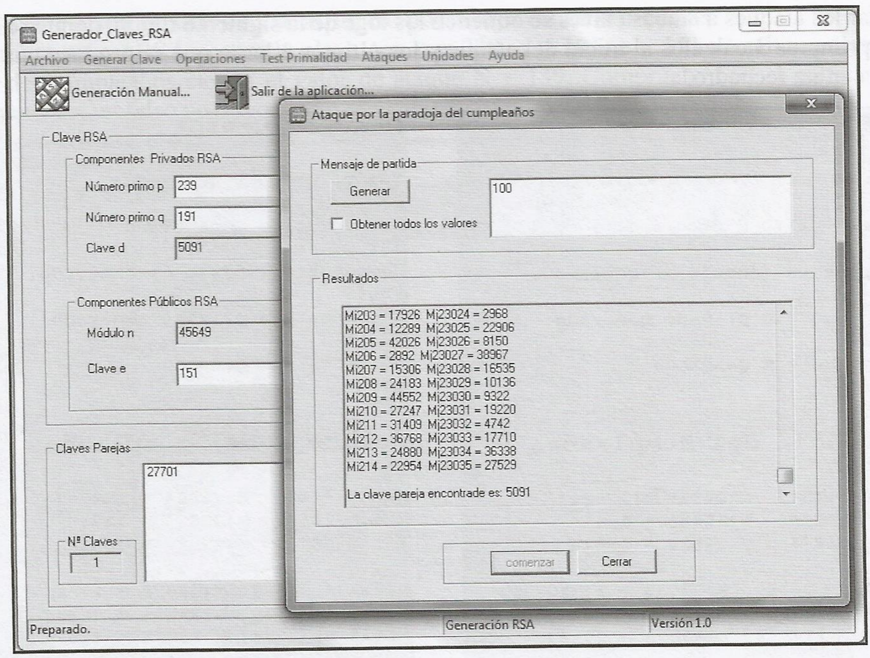


Figura 65. Solución: Ejercicio – Ataque cumpleaños para clave generada con e=151.

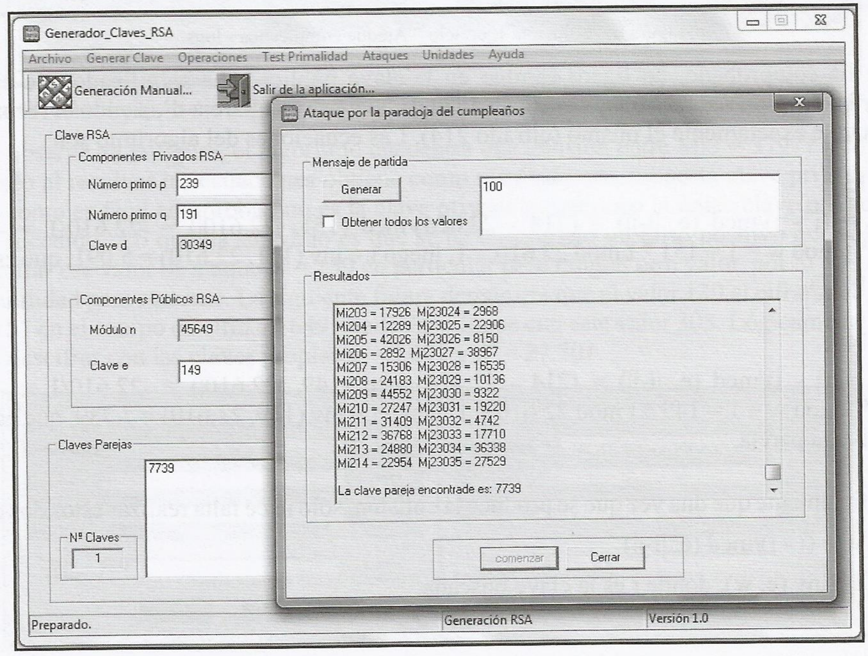


Figura 66. Solución: Ejercicio – Ataque cumpleaños para clave generada con e=149.

Al realizar los ataques a ambas claves se obtienen los logs de la siguiente figura, donde se muestran sólo los primeros 3 cálculos al inicio del ataque y los últimos 3 hasta que se produce una colisión, marcada con un recuadro.

```
CLAVE GENERADA

Número primo P generado:
239
Número primo Q generado:
191
Componente (módulo n) generado:
45649
Componente d privado generado:
5091
Componente e generado:
151

Paradoja del cumpleaños

Mi3 = 41371   Mj22824 = 22954
Mi4 = 28690   Mj22825 = 12950
Mi5 = 38762   Mj22826 = 16828
... ... ... ... ... ... ... ..
Mi212 = 36768   Mj23033 = 17710
Mi213 = 24880   Mj23034 = 36338
Mi214 = 22954   Mj23035 = 27529

La clave pareja encontrada es: 5091
```

```
CLAVE GENERADA

Número primo P generado:
239
Número primo Q generado:
191
Componente (módulo n) generado:
45649
Componente d privado generado:
30349
Componente e generado:
149

Paradoja del cumpleaños

Mi3 = 41371   Mj22824 = 22954
Mi4 = 28690   Mj22825 = 12950
Mi5 = 38762   Mj22826 = 16828
... ... ... ... ... ... ... ..
Mi212 = 36768   Mj23033 = 17710
Mi213 = 24880   Mj23034 = 36338
Mi214 = 22954   Mj23035 = 27529

La clave pareja encontrada es: 7739
```

Figura 67. Solución: Ejercicio – Ataque cumpleaños - logs.

Como se ha comprobado, un ligero cambio en la clave pública e ha significado que en vez de encontrar la clave privada d se haya encontrado una clave privada pareja d', siendo el recorrido que hace el ataque exactamente el mismo (cifrado 214). Las ecuaciones del algoritmo son:

Para e = 151:

> w = (i - j)/mcd (e, |i-j|) = (214 - 22.824)/mcd (151, |22.610|) = -22.610/1 = - 22.610. e * t mod w = 1 = 151 * t mod 22.610 = 1, luego t = inv (151, 22.610) = 5.091, que es la clave privada.

Para e = 149:

> w = (i - j)/mcd (e, |i-j|) = (214 - 22.824)/mcd (149, |22.610|) = -22.610/1 = - 22.610. e * t mod w = 1 = 149 * t mod 22.610 = 1, luego t = inv (149, 22.610) = 7.739, que es la clave privada pareja.

Destacar finalmente que una vez que se produce la colisión, sólo hace falta realizar estos dos cálculos:

1) w = (i - j)/mcd (e, |i-j|)
2) t = inv (e, w), donde t es la clave buscada

Ejercicio 5. Falsos positivos y ataque de paradoja de cumpleaños

Con el programa genRSA genere la clave con p = 239, q = 191, e = 151 y realice un ataque por paradoja del cumpleaños siendo el valor de ataque N = 139. Compruebe con el software que el ataque encuentra un falso positivo.

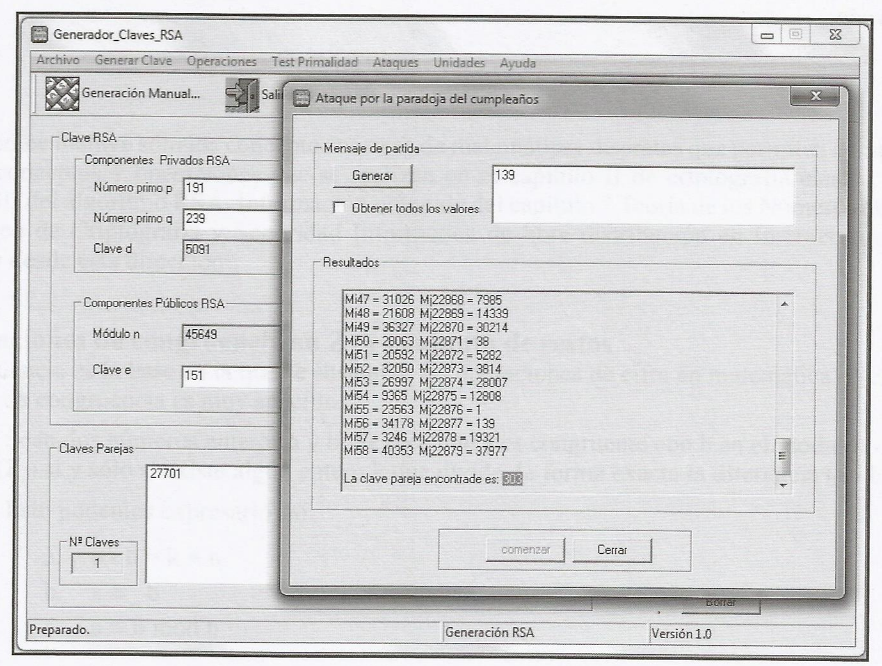


Figura 68. Solución: Ejercicio - Falsos positivo 303 al realizar el ataque de paradoja de cumpleaños con N=139.

Como se observa en la figura, el algoritmo encuentra una colisión entre Mi3 = 37977 y Mj22879 = 37977, pero al resolver las ecuaciones obtiene como resultado una supuesta clave privada de valor 303 que, como es fácil comprobar, no es la clave privada ni tampoco la única clave privada pareja. ¿Qué ha sucedido? Lo que ha sucedido es que se ha encontrado un falso positivo que sólo servirá para descifrar ese valor de ataque N = 139 en particular pero ningún otro valor, por lo que no tendrá ninguna utilidad para atacante. La siguiente figura demuestra que el valor 139 al cifrarse con la clave pública 151 en el cuerpo de cifra 45.649 puede descifrarse con este valor 303. Lógicamente, también se puede descifrar con las claves propias d = 5.091 y d' = 27.701.

Figura 69. Cálculo de exponenciación modular.

A algo similar va a llegar si se usa como valor de ataque N alguno de los 33 números no cifrables de esta clave que, a excepción del 0 y del 1, son: 240, 955, 956, 1195, 1911, 2150, 5737, 6692, 7410, 7647, 8365, 9320, 11951, 12906, 13861, 31788, 32743, 33698, 36329, 37284, 38002, 38239, 38957, 39912, 43499, 43738, 44454, 44693, 44694, 45409, 45648. Compruébelo y saque conclusiones de lo observado..

Apéndice A.
Fundamentos de Matemáticas Discretas

Este apéndice incluye sólo los conceptos básicos de matemáticas discretas que permiten comprender algunos conceptos y operaciones que se realizan en el capítulo II de criptografía clásica y en el capítulo III del algoritmo RSA. Información obtenida del capítulo 7 Teoría de los Números del Libro Electrónico de Criptografía y Seguridad Informática de libre distribución en Internet que puede descargar desde esta dirección[1].

1. Operaciones de congruencia en Z_n y conjunto de restos

La congruencia es la base en la que se sustentan las operaciones de cifra en matemática discreta. El concepto de congruencia es muy sencillo:

1) Sean dos números enteros a y b. Se dice que a es congruente con b en el módulo o cuerpo n (Z_n) si y sólo si existe algún entero k que divide de forma exacta la diferencia (a - b).

2) Esto podemos expresarlo así:

a. $a - b = k * n$

b. $a \equiv_n b$

c. $a \equiv b \bmod n$

Por lo tanto, ante la pregunta si 18 congruente con 3 módulo 5 la respuesta es afirmativa puesto que:

$18-3 = 15 = k*5$ con $k = 3$

Estas operaciones modulares se usan habitualmente en criptografía, tanto clásica como moderna, si bien en ambos casos el concepto de módulo o cuerpo de cifra -normalmente indicado como n salvo casos especiales- es muy distinto. En criptografía clásica el módulo de cifra n se corresponde con el tamaño del alfabeto utilizado, por ejemplo 27 si el alfabeto contempla sólo las letras mayúsculas del castellano, mientras que en los sistemas de cifra modernos no tiene ninguna relación con el alfabeto, en este caso ASCII.

Sin entrar en detalles matemáticos, en criptografía es normal indicar la congruencia del ejemplo anterior como 18 mod 5 = 3 donde 3 será el resto o residuo.

El conjunto de números que forman los restos dentro de un cuerpo Z_n será muy importante en criptografía.

---

1 Libro Electrónico de Criptografía y Seguridad Informática, V4.1, Jorge Ramió, 2006.
http://www.criptored.upm.es/guiateoria/gt_m001a.htm

2. Conjunto completo de restos CCR

Para cualquier entero positivo n, el conjunto completo de restos será CCR = {0, 1, 2, ... n-1}, es decir:

$$\forall a \in Z \quad \exists ! r_i \in CCR / a \equiv r_i \bmod n$$

Por tanto, CCR (11) = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}, CCR (6) = {0, 1, 2, 3, 4, 5}

También se podría representar, por ejemplo, el segundo conjunto como CCR (6) = {12, 7, 20, 9, 16, 35}. En este caso ambos conjuntos son equivalentes puesto que en módulo 6 se tiene:

12 mod 6 = 0; 7 mod 6 = 1; 20 mod 6 = 2; 9 mod 6 = 3; 16 mod 6 = 4; 35 mod 6 = 5

Es más, podría incluso usarse números negativos. Por ejemplo en este módulo 6 en vez de 20, cuyo resto es 2, se puede poner -10 en tanto - 10 mod 6 = 2

Sin embargo lo común es operar con restos dentro de la zona llamada canónica, esto es entre 0 y n-1.

3. El conjunto reducido de restos

El conjunto reducido de restos, conocido como CRR de n, es el subconjunto {0, 1, ... n_i, ... n-1} de restos, primos con el grupo n.

Si n es primo, todos los restos serán primos con él. Como el cero no es una solución, entonces:

$$CRR = \{1, ..., n_i, ... n-1\} \ / \ mcd\ (n_i, n) = 1$$

Por ejemplo, CRR mod 7 = {1, 2, 3, 4, 5, 6}, CRR (15) = {1, 2, 4, 7, 8, 11, 13, 14}

¿Qué utilidad tiene esto en criptografía?

El conocimiento del CRR permitirá aplicar un algoritmo para el cálculo del inverso multiplicativo de un número *x* dentro de un cuerpo *n* a través de la función $\phi(n)$, denominada Función de *Euler* o Indicador de *Euler*.

4. La Función de Euler $\phi(n)$

El Indicador o Función de *Euler* $\phi(n)$ entregará el número de elementos del CRR. Puede representarse de cuatro formas:

1) Si n es un número primo p.

2) Si n se representa como $n = p^k$ con p primo y k entero.

3) Si n es el producto n = p*q con p y q primos.

4) Si n es un número cualquiera, forma genérica:

$$n = p_1^{e1} * p_2^{e2} * ... * p_t^{et} = \Pi_{\text{(desde i=1 hasta t)}}\, p_i^{ei}$$

Para este libro sólo interesan los casos 1 y 3.

1. Función $\phi(n)$ de *Euler* cuando n = p

Si n es primo, $\phi(n)$ será igual a CCR menos el 0, es decir $\phi(p) = p - 1$.

Si n es primo, entonces CRR = CCR - 1 ya que todos los restos de n, excepto el cero, serán primos entre sí.

CRR(7) = {1, 2 ,3, 4, 5, 6} seis elementos porque $\phi(7) = p - 1 = 7 - 1 = 6$

CRR(11) = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} 10 elementos porque $\phi(11) = p - 1 = 11 - 1 = 10$

2. Función $\phi(n)$ de *Euler* cuando n = p*q

Si n = p*q (con p y q primos), $\phi(n) = \phi(p*q) = \phi(p)*\phi(q) = (p-1)(q-1)$

Explicación: de los p*q elementos del CCR, se quitan todos los múltiplos de p = 1*p, 2*p, ... (q - 1)*p, y también todos los múltiplos de q = 1*q, 2*q, ... (p - 1)*q, además del cero. Por lo tanto:

$\phi(p*q) = p*q - [(q-1) + (p-1) +1] = p*q - q - p + 1 = (p-1)(q-1)$

Esta será una de las operaciones más utilizadas en criptografía pues es el cuerpo trampa del sistema RSA que permite el cálculo de la clave privada a partir de la pública.

5. Inversos en un cuerpo

En criptografía deberá estar permitido invertir una operación para deshacer un cifrado, esto es realizar la operación de descifrado que permita recuperar el texto en claro.

Aunque la cifra es una función, en lenguaje coloquial la operación de cifrado podría interpretarse como una "multiplicación" y la operación de descifrado como una "división", si bien en este caso se habla de una multiplicación por el inverso. Esta analogía sólo será válida en el cuerpo de los enteros Z_n con inverso.

Luego, si en una operación de cifra la función es el valor a dentro de un cuerpo n, deberemos encontrar el inverso de a en dicho cuerpo n, a^{-1} mod n, para descifrar.

En otras palabras, si $a * x \equiv 1 \bmod n$, se dice que x es el inverso multiplicativo de a en Z_n y se denotará por a^{-1} y la ecuación del inversos se indicará como: inv (a, n) = x.

Hay que tener en cuenta estas dos características:

1) No siempre existen el inverso de un elemento en Z_n. Por ejemplo, si n = 6, en Z_6 no existe el inverso del resto 2, pues la ecuación $2*x \equiv 1 \bmod 6$ no tiene solución.

2) Si n es un número primo p, entonces todos los elementos de Z_p salvo el cero tienen inverso. Por ejemplo, en Z_5 se tiene que: $1^{-1} \bmod 5 = 1$; $2^{-1} \bmod 5 = 3$, $3^{-1} \bmod 5 = 2$; $4^{-1} \bmod 5 = 4$.

Si existe una primalidad relativa entre a y n en la expresión inv (a, n) entonces el inverso x buscado existe. Por ejemplo inv (6, 13) = x existe porque 13 es un número primo y todos los elementos o restos del primo son primos con el mismo, aunque no se conozca aún el valor de x, en cambio inv (10, 35) = x no existe porque 10 y 35 tiene un factor en común, el número 5.

$\exists$ inverso a^{-1} en mod n *ssi* mcd (a, n) = 1

Si mcd (a, n) = 1, el resultado de a*i mod n (para i todos los restos de n) serán valores distintos dentro del cuerpo n y sólo uno de ellos será igual a 1 que entrega el inverso buscado.

$$\text{mcd}(a, n) = 1 \quad \Rightarrow \quad \exists\, x\,!\ \ 0 < x < n \ / \ a * x \bmod n = 1$$

Ejemplo: Sea: a = 4 y n = 9. Ni a ni n son primos; no tienen factores en común porque $4 = 2^2$ y $9 = 3^3$.

Valores de i = {1, 2, 3, 4, 5, 6, 7, 8}

4*1 mod 9 = 4 4*2 mod 9 = 8 4*3 mod 9 = 3 4*4 mod 9 = 7 4*5 mod 9 = 2

4*6 mod 9 = 6 **4*7 mod 9 = 1** 4*8 mod 9 = 5

Como se observa marcado en negrita, 4 será el inverso de 7 en módulo 9 y el valor 7 será el inverso de 4 en ese módulo 9, pues 4*7 mod 9 = 28 mod 9 = 3*9 + 1 mod 9 = 1.

Y esto podría indicarse así: 4 = (1/7) mod 9 o bien 7 = (1/4) mod 9.

Inversos aditivos y multiplicativos

(A+B) mod 5

B +	0	1	2	3	4
A 0	0	1	2	3	4
1	1	2	3	4	**0**
2	2	3	4	**0**	1
3	3	4	**0**	1	2
4	4	**0**	1	2	3

(A*B) mod 5

B *	0	1	2	3	4
A 0	0	0	0	0	0
1	0	**1**	2	3	4
2	0	2	4	**1**	3
3	0	3	**1**	4	2
4	0	4	3	2	**1**

En la operación suma siempre existirá el inverso o valor identidad de la adición (0) para cualquier resto del cuerpo. Su valor es único. En la operación producto, de existir un inverso o valor de identidad de la multiplicación (1), éste es único y la condición para ello es que el número y el módulo sean primos entre sí.

6. El Teorema de Euler

El teorema de *Euler* dice que mcd (a, n) = 1, entonces $a^{\phi(n)} \bmod n = 1$.

Igualando a*x mod n = 1 con $a^{\phi(n)} \bmod n = 1$, se tiene:

$$a^{\phi(n)} * a^{-1} \bmod n = x \bmod n$$

$$x = a^{\phi(n)-1} \bmod n$$

El valor x será el inverso de a en el cuerpo n.

Observe que se ha 'dividido' por *a* en el cálculo anterior (a^{-1}). Esto se puede hacer porque mcd (a, n) = 1 y por lo tanto hay un único valor inverso en el cuerpo n que lo permite.

¿Cuál será es el inverso de 4 en módulo 9?

$x = \text{inv}(4, 9) = 4^{\phi(9)-1} \bmod n$

Como $\phi(9) = 6$ entonces $x = 4^{6-1} \bmod 9 = 4^5 \bmod 9 = 1.024 \bmod 9 = 7$

7. Pequeño teorema de Fermat

Si el cuerpo de trabajo es un primo p, entonces mcd (a, p) = 1 y $a^{\phi(p)}$ mod p = 1

Entonces realizando las mismas operaciones anteriores:

$$a^{\phi(p)} * a^{-1} \bmod p = x \bmod p$$
$$x = a^{p-2} \bmod p$$

Realizar el cálculo a^i mod n cuando el valor de i y el de a son grandes, se hace tedioso y significa tiempo de cómputo.

Si no se conoce $\phi(n)$ o no se desea usar los teoremas de *Euler* o *Fermat*, siempre puede encontrar el inverso de a en el cuerpo n usando el Algoritmo Extendido de *Euclides*.

8. Algoritmo Extendido de Euclides (AEE)

Si mcd (a, n) = 1 y a*x mod n = 1, entonces x = inv (a, n). Luego podemos escribir:

$n = C_1*a + r_1$ Con $a > r_1$

$a = C_2*r_1 + r_2$ Con $r_1 > r_2$

$r_1 = C_3*r_2 + r_3$ Con $r_2 > r_3$

... ...

$r_{n-2} = C_n*r_{n-1} + 1$ Con $r_{n-1} > 1$

$r_{n-1} = C_{n+1}*1 + 0$

Concluye el algoritmo

En el paso "$r_{n-2} = C_n*r_{n-1} + 1$ Con $r_{n-1} > 1$" si ahora se vuelve hacia atrás desde este valor unitario, se obtiene el inverso de a en el cuerpo n. El siguiente ejemplo lo muestra:

Tabla de restos del AEE

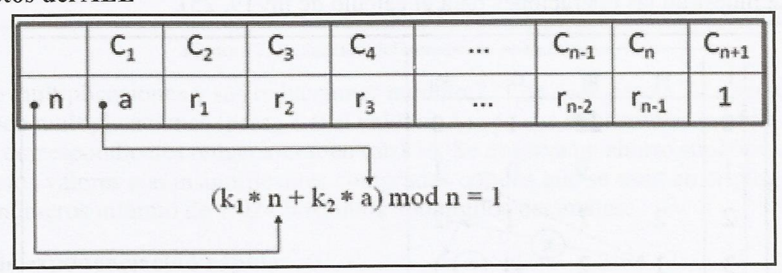


Figura 70: Tabla de restos del AEE.

Ordenando por restos desde el valor 1 se llega a una expresión del tipo $(k_1 * n + k_2 * a) \bmod n = 1$, en donde el inverso de a en n lo dará el coeficiente k_2 puesto que $k_1 * n \bmod n = 0$.

Ejemplo: encontrar el inv (9, 25) por el método de restos de *Euclides*

a) 25 = 2*9 + 7 Restos: 7 = 25 - 2*9 7 = 25 - 2*9

b) 9 = 1*7 + 2 Restos: 2 = 9 - 1*7 2 = 9 - 1*(25 - 2*9) = 3*9 - 1*25

c) 7 = 3*2 + 1 Restos: 1 = 7 - 3*2 1 = (25 - 2*9) - 3*(3*9 - 1*25) = 4*25 - 11*9

d) 2 = 2*1 + 0 1 = 4*25 - 11*9 mod 25 = - 11*9 mod 25

Pues 4*25 mod 25 = 0

Como 1 = - 11*9 mod 25, se obtiene que inv (9, 25) = -11

Como el -11 es un conjunto de equivalencia se deja dentro de la parte canónica sumando el módulo 25 tantas veces como sea necesario, en este caso sólo una: - 11 + 25 = 14.

Por tanto inv (9, 25) = 14 que se comprueba: 14*9 mod 25 = 126 mod 25 = 5*25 + 1 mod 25 = 1

Cálculo de inversos mediante AEE

Para encontrar x = inv (A, B)

Hacer $(g_0, g_1, u_0, u_1, v_0, v_1, i) = (B, A, 1, 0, 0, 1, 1)$

Mientras $g_i \neq 0$ hacer

Hacer y_{i+1} = parte entera (g_{i-1}/g_i)

Hacer $g_{i+1} = g_{i-1} - y_{i+1} * g_i$

Hacer $u_{i+1} = u_{i-1} - y_{i+1} * u_i$

Hacer $v_{i+1} = v_{i-1} - y_{i+1} * v_i$

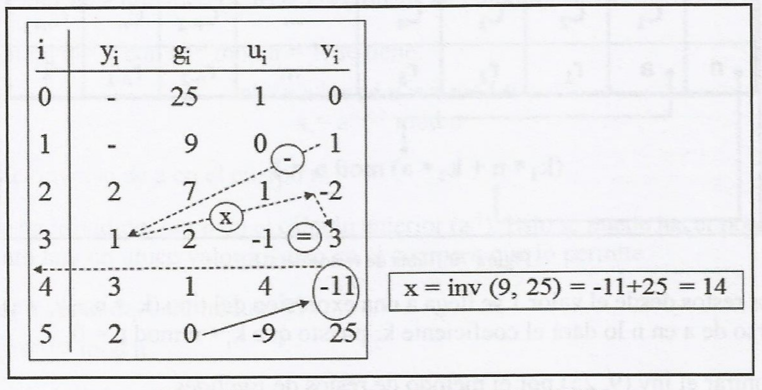
Hacer i = i+1

Si $(v_{i-1} < 0)$

Hacer $v_{i-1} = v_{i-1} + B$

Hacer $x = v_{i-1}$

En la figura se muestran las operaciones para el cálculo de inv (9, 25).



i	y_i	g_i	u_i	v_i
0	-	25	1	0
1	-	9	0	1
2	2	7	1	-2
3	1	2	-1	3
4	3	1	4	-11
5	2	0	-9	25

Figura 71: operaciones para el cálculo de inv (9, 25).

9. Exponenciación rápida

En criptografía se necesitará realizar operaciones computacionalmente costosas, generalmente de la forma A^B mod n, en donde el módulo n será por general 2.048 ó 1.024 bits, el exponente B un valor de 17 bits en el intercambio de clave o del mismo orden que n en la firma digital, y la base A de unos cientos de bits en ambas operaciones. Es necesario entonces acelerar esta operación.

1) En la operación A^B mod n, se representa el exponente B en binario.

2) Se calculan los productos $A^{2^\wedge j}$ con j = 0 hasta n-1, siendo n el número de bits que representan el valor B en binario.

3) Sólo se toman en cuenta los productos en los que en la posición j del valor B en binario aparece un 1.

Por ejemplo si se pide calcular x = 12^{37} mod 221 = 207, 12^{37} es un número para nosotros algo grande, de 40 dígitos: 8505622499821102144576131684114829934592, que no entra en una calculadora "clásica". Pero sin embargo esta operación se sigue pudiendo hacer usando el principio de reducibilidad.

Si se usa reducibilidad, es decir 12^2 mod 221, su resultado se multiplica por 12 y se reduce nuevamente mod 221, y así hasta todas las iteraciones que marca el exponente, se deberían realizar 36 operaciones de multiplicación más sus correspondientes reducciones a módulo, 72 operaciones.

Si por el contrario se usa un método de exponenciación rápida como el que se indicará, se reduce significativamente el número de operaciones como muestra la figura.

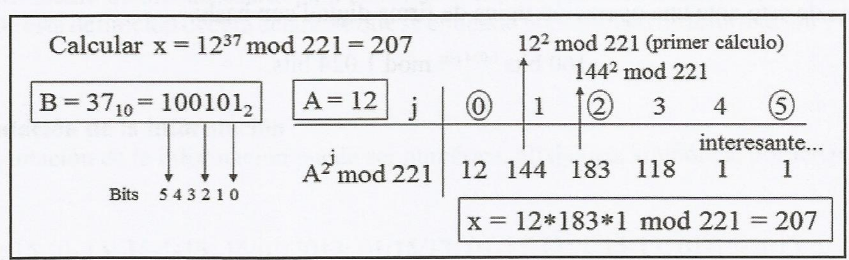


Figura 72: Reducción del número de operaciones.

En vez de 36 multiplicaciones y sus reducciones módulo 221 en cada paso... 72 operaciones, se han realizado cinco multiplicaciones (para j = 0 el valor es A) con sus reducciones módulo 221, más dos al final y sus correspondientes reducciones; en total 14. Se observa un ahorro superior al 80% pero el tamaño de estos valores son insignificantes comprados con los que se usan en criptografía de clave pública con números mínimo de 1.024 bits, unos 300 dígitos decimales.

Algoritmo de exponenciación rápida
Hallar x = A^B mod n

Obtener representación binaria del exponente B de k bits:

$B_2 \rightarrow b_{k-1}b_{k-2}...b_i...b_1b_0$

Hacer x = 1

Para i = k-1, ..., 0 hacer

$x = x^2 \bmod n$

Si (b_i = 1) entonces x = x*A mod n

Ejemplo: calcule 19^{83} mod 91

$83_{10} = 1010011_2 = b_6b_5b_4b_3b_2b_1$

x = 1

i=6 b_6=1 $x = 1^2*19 \bmod 91 = 19$ x = 19

i=5 b_5=0 $x = 19^2 \bmod 91 = 88$ x = 88

i=4 b_4=1 $x = 88^2*19 \bmod 91 = 80$ x = 80

i=3 b_3=0 $x = 80^2 \bmod 91 = 30$ x = 30

i=2 b_2=0 $x = 30^2 \bmod 91 = 81$ x = 81

i=1 b_1=1 $x = 81^2*19 \bmod 91 = 80$ x = 80

i=0 b_0=1 $x = 80^2*19 \bmod 91 = 24$ x = 24

19^{83} mod 91 = 24

19^{83} = 1,3694585098795051015573767467l8e+106

En este caso se han realizado sólo 16 operaciones frente a 164 por reducibilidad. Piense ahora la importancia de esto ante una operación típica de firma digital con hash:

160 bits $^{1.024 \text{ bits}}$ mod 1.024 bits.

Apéndice B
Teoría de la información

En este apéndice se incluyen sólo los conceptos básicos de la teoría de la información con algunas nociones de lo propuesto por *Claude Shannon* al respecto y que permiten comprender conceptos y operaciones que se realizan en el capítulo II de este libro sobre criptografía clásica. Información obtenida del capítulo 6 Teoría de la Información del Libro Electrónico de Criptografía y Seguridad Informática de libre distribución en Internet que puede descargar desde esta dirección[1].

1. ¿Qué es la teoría de la información?

Definición de información:
Es el conjunto de datos o mensajes inteligibles creados con un lenguaje de representación y que debemos proteger ante las amenazas del entorno, durante su transmisión o almacenamiento, usando técnicas criptográficas entre otras herramientas.

Y la teoría de la información medirá la *cantidad de información* que contiene un mensaje a través del número medio de bits necesario para codificar todos los posibles mensajes con un *codificador óptimo*. De esta definición deberá aclararse qué se entiende por cantidad de información y codificador óptimo.

Representación de la información
La representación de la información puede ser numérica, alfabética, simbólica, por lenguaje, etc.

Ejemplos:

15/01/13; 15-01-13; 15-1-13; 15/01/2013; 01/15/13; 01-15-13; 1-15-13; 01-15-2013 ...

Todas las representaciones y otras más con distintas posiciones de dd/mm/aa anteriores se interpretan como el 15 de enero del año 2013.

Vitaminas: B_{12}, C, ...
Grupo sanguíneo: A2 Rh+ ...
Elementos: Fe, Si, Hg ...
Compuestos químicos: H_2O, CO_2 ...

Lo más común es a través de un lenguaje con código, por ejemplo una frase "Cuando tenga 18 años seré mayor de edad".

---

1 Libro Electrónico de Criptografía y Seguridad Informática, V4.1, Jorge Ramió, 2006.
http://www.criptored.upm.es/guiateoria/gt_m001a.htm

Mensajes que contienen información

La recepción de un mensaje puede entregarnos más o menos información dependiendo del contexto en que nos encontremos. Esto puede analizarse:

a) En función de la *extensión* del mensaje recibido.

b) En función de la *utilidad* del mensaje recibido.

c) En función de la *sorpresa* del mensaje recibido.

d) Dependiendo del *entorno* de esa sorpresa.

e) En función de la *probabilidad* de recibir un mensaje.

Este último enfoque de la probabilidad de recibir un mensaje u otro, orientado a la ingeniería y usado por *Claude Shannon* en su estudio, es el que aquí interesa.

Incertidumbre e información

Ante varios mensajes posibles, en principio todos equiprobables, aquel que tenga una menor probabilidad de aparición será el que contenga una mayor cantidad de información.

Concepto de variable aleatoria

Sea X una variable aleatoria con n estados posibles con $X = x_i$ una ocurrencia iésima:

$X = \{x_1, x_2, x_3, ..., x_{n-1}, x_n\}$

$p_1 = p(x_1), p_2 = p(x_2), ..., p_n = p(x_n)$

Como: $0 \leq p_i \leq 1$ para $i = 1, 2, ..., n$

Entonces:

$$\sum_{i=1}^{n} p_i = 1$$

La probabilidad de que ocurra p_1 o p_2 o p_3, etc. será siempre la unidad porque seguro será uno de ellos.

Definición de cantidad de información

Definiremos c_i a la cantidad de información del estado i, como el logaritmo en base dos de la probabilidad de que ocurra el estado iésimo.

$$c_i = - \log_2 (p_i)$$

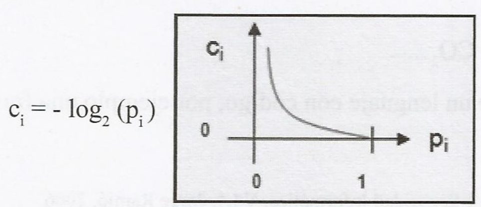


Figura 73: Gráfica correspondiente a $c_i = - \log_2 (p_i)$.

Logaritmo: $p(x_i) = 1 \Rightarrow$ no hay incertidumbre: $c_i = 0$
$p(x_i) = 0 \Rightarrow$ máxima incertidumbre: $c_i \rightarrow \infty$

Signo: $p(x_i) < 1 \Rightarrow \log p(x_i)$ será negativo

Base 2: Un fenómeno binario $\Rightarrow$ dos estados (bit)

Grado de indeterminación

$$c_i = \frac{\text{Grado de indeterminación previo}}{\text{Grado de indeterminación posterior}}$$

Ejercicio. En una bolsa hay:

- Un papel con un círculo blanco
- Un papel con un círculo negro
- Un papel con un cuadrado blanco
- Un papel con un cuadrado negro
- Un papel con un triángulo blanco
- Un papel con un triángulo negro

Si una persona saca de la bolsa tres papeles cualesquiera, la pregunta es cómo adivinar con el menor esfuerzo posible qué combinación tiene en la mano.

Como p(xi) = 1/8 entonces la Incertidumbre inicial es $I_i = 8$

Si se entregan algunas pistas, se tiene:

a) Pista 1: Las figuras no son del mismo color:

Resultado: la I_i baja de 8 a 6 al descartarse las 2 combinaciones de figuras todos del mismo color

b) Pista 2: El círculo es blanco:

Resultado: la I_i baja de 6 a 3 al descartarse las 3 combinaciones que quedan con el círculo negro.

c) Pista 3: Hay dos figuras blancas:

Resultado: la I_i baja de 3 a 2 al descartarse la combinación con dos figuras negras.

d) Pista 4: El cuadrado es negro:

Resultado: la I_i baja de 2 a 1 al descartarse la combinación con cuadrado blanco.

Se acaba la incertidumbre pues la solución es: círculo blanco, cuadrado negro, triángulo blanco.

Matemáticamente:

a) Las figuras no son del mismo color. I_i baja de 8 a 6: $c_{i1} = \log (8/6) = \log 8 - \log 6$

b) El círculo es blanco. I_i baja de 6 a 3: $c_{i2} = \log (6/3) = \log 6 - \log 3$

c) Hay dos figuras blancas. I_i baja de 3 a 2: c_{i3} = log (3/2) = log 3 - log 2

d) El cuadrado es negro. I_i baja de 2 a 1: c_{i4} = log (2/1) = log 2 - log 1

Como todas las magnitudes se pueden sumar como escalares: $c_i = c_{i1} + c_{i2} + c_{i3} + c_{i4}$ = log 8 - log 1 = log 8

La base del logaritmo

Sean

I_i la indeterminación inicial

I_f la indeterminación final

$c_i = \log (I_i / I_f) = \log I_i - \log I_f$

La cantidad de información tiene como unidad de medida la de un fenómeno de sólo dos estados, un fenómeno binario. Luego:

$c_i = \log_b (2/1) = \log_b 2 - \log_b 1$

Si $\log_b 2$ debe ser igual a 1 entonces la base b = 2.

Precisamente a esta unidad se le llama bit (<u>bi</u>nary digi<u>t</u>)

En el ejemplo anterior: $c_i = \log_2 8 = 3$. Es decir, se pasa de la incertidumbre total a la certeza con sólo 3 preguntas.

Con sólo tres preguntas "*más o menos inteligentes*" podemos pasar de la incertidumbre total a la certeza. Como hay 8 opciones y se asume que la elección es la 3:

Pregunta 1: ¿Está entre la opción 1 y la 4? ⇒ Sí

Pregunta 2: ¿Está entre la opción 1 y la 2? ⇒ No

Pregunta 3: ¿Es la opción 4? ⇒ No

Se acaba de la indeterminación con sólo tres preguntas; es la opción 3

2. Entropía de los mensajes, ratio y redundancia del lenguaje

Si un fenómeno tiene un grado de indeterminación k y sus estados son equiprobables, la probabilidad p de que se dé uno de esos estados será 1/k. Luego:

$$c_i = \log_2 (k/1) = \log_2 [1/(1/k)] = - \log_2 p$$

Si ahora cada uno de estos estados tiene una probabilidad distinta p_i, la entropía H será igual a la suma ponderada de la cantidad de información:

$$H = - p_1 \log_2 p_1 - p_2 \log_2 p_2 - \ldots - p_k \log_2 p_k$$

$$H = - \sum_{i=1}^{k} p_i \log_2 p_i$$

La entropía de un mensaje X, que se representa por H(X), es el *valor medio ponderado* de la cantidad de información de los diversos estados del mensaje.

$$H(X) = - \sum_{i=1}^{k} p(x_i) \log_2 p(x_i)$$

Es una medida de la *incertidumbre media* acerca de una variable aleatoria y el *número de bits de información.*

De la definición anterior, después del ejemplo de los papeles, se ha aclarado el concepto de incertidumbre en H. Lo que es necesario aclarar más adelante es lo del número de bits de información.

Propiedades de la entropía

1) La entropía es no negativa y se anula si y sólo si un estado de la variable es igual a 1 y el resto 0. Esta demostración es sencilla.

2) La entropía será máxima, hay mayor incertidumbre del mensaje, cuando exista una equiprobabilidad en todos los valores de la variable X. La demostración empírica es muy fácil; no obstante la demostración matemática de este máximo no es directa. El valor máximo de H(X) para una variable de n estados será $\log_2 n$.

Si hay n estados equiprobables, entonces $p_i = 1/n$.

$$H(X) = - \Sigma\, p_i \log_2 p_i = - n(1/n) \log_2 (1/n) = - (\log_2 1 - \log_2 n)$$

Como $\log_2 1 = 0$ entonces $H(X)_{máx} = \log_2 n$

Entropía condicional o equivocación de X

Si existe una segunda variable Y que influya sobre X, esto entregará importante información adicional:

$$H(X/Y) = - \sum_{x,y} p_{(x,y)} \log_2 p_{(x,y)}$$

Donde p(x,y) = p(y)p(x/y) y la relación p(x/y) es la probabilidad de que se obtenga un estado X conocido el valor de Y. Luego:

$$H(X/Y) = - \sum_{y} p_{(y)} \sum_{x} p_{(x/y)} \log_2 p_{(x/y)}$$

Codificador óptimo

Introduciendo el signo negativo dentro del logaritmo en la expresión de la entropía, ésta nos quedará como:

$$H(X) = \Sigma\; p(x) \log_2 [1/p(x)]$$

La expresión $\log_2 [1/p(x)]$ representará el número necesario de bits para codificar el mensaje X en un codificador óptimo. Codificador óptimo es aquel que para codificar un mensaje X usa el menor número posible de bits.

Codificación con el método de Huffman

La siguiente figura muestra una posible codificación óptima con este método para el siguiente mensaje:

M = MI MAMÁ ME MIMA (mensaje con varios letras repetidas para que se vea mejor el efecto)

1) Se contabilizan los caracteres, 15 en este caso, obteniéndose los valores que se muestran en la figura con la letra M repetida 6 veces.

2) Se comienza a armar el árbol de izquierda a derecha desde los caracteres menos frecuentes a los más frecuentes y se indica en el nodo superior la cantidad de ocurrencias hasta ese momento.

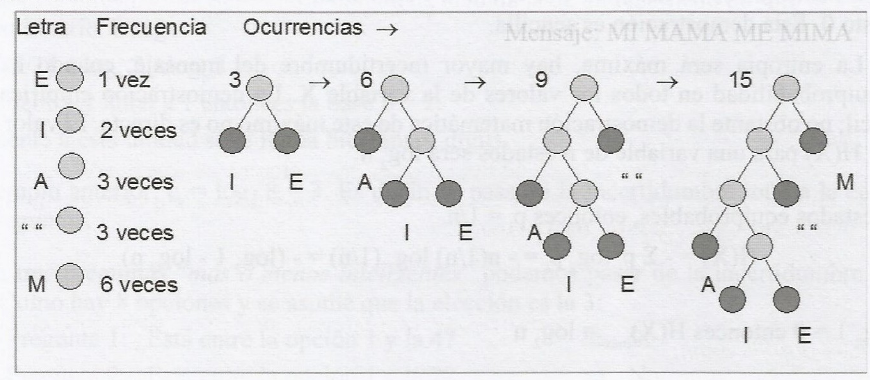


Figura 74: Codificación óptima con el método de Huffman para un mensaje en concreto.

Una codificación desde el nodo superior hacia abajo podría ser:

M = 1; “ ” = 01; A = 000; I = 0010; E = 0011

Por tanto, el mensaje M de 15 caracteres ASCII (15*8 = 120 bits) se queda en 33:

M = 1 0010 01 1 000 1 000 01 1 0011 01 1 0010 1 000 (33 bits)

El número necesario de bits y la entropía

Ejemplo: se calculará el número de bits óptimo para la codificación del siguiente mensaje de 8 caracteres M = LELA ELLA (mensaje especial para que con pocos caracteres se observe este efecto).

Solución: p(L) = 0,5; p(E) = 0,25; p(A) = 0,25; y obviamente Σ p(L, E, A) = 1,0.

Para codificar L necesitaremos 1 bit: $\log_2 [1/ P(L)] = \log_2 2 = 1$

Para codificar E necesitaremos 2 bits: $\log_2 [1/ P(E)] = \log_2 4 = 2$

Para codificar A necesitaremos 2 bits: $\log_2 [1/ P(A)] = \log_2 4 = 2$

Luego, si L se codifica como 0, E como 10 y A como 11, el mensaje M se codificará como: 0 10 0 11 10 0 0 11, es decir se transmiten 12 bits.

Si se calcula la entropía de M se obtendrá H(M) = 1,5 y al mismo valor se llega con el concepto de número medio de bits: para codificar un mensaje M de 8 elementos, hemos usado 12 bits. Luego 12/8 = 1,5 bits por elemento.

La ratio r del lenguaje

Es el número de "bits de información" en cada carácter para mensajes con una longitud igual a N caracteres. Luego, según la definición de entropía, se tiene:

$$r = H(X)/N \quad \text{(bits/letra)}$$

Si se codifica un mensaje letra a letra, suponiendo además equiprobabilidad entre estas letras, se obtiene la denominada ratio absoluta del lenguaje, R:

$$R = H(X)$$

Por lo tanto si se trabaja con sólo 27 letras: $R_{castellano} = \log_2 n = \log_2 27 = 4{,}75$ (bits/letra)

Ratio verdadera del lenguaje

Como las letras que aparecen en un texto no tienen igual probabilidad, su frecuencia de aparición es distinta, los lenguajes está muy estructurados, hay bloques de dos palabras (digramas) característicos, trigramas, poligramas, etc., la ratio baja mucho...

$$1{,}2 < r < 1{,}5$$

A este valor se llega codificando los mensajes con monogramas, digramas, trigramas, etc., según el estudio hecho por *Shannon*.

¿Qué significa esto? Si un alfabeto consta de L elementos existirán 2^{R*N} mensajes posibles de longitud N, la entropía máxima será $H(X)_{máx} = \log_2 L$, y sólo habrá 2^{r*N} mensajes que tengan sentido.

Esto no significa que se pueda codificar todos los mensajes de 27 caracteres con sólo 2 bits (esto sería imposible), sólo significa que la información que contiene cada letra es tan sólo de 1,5 bits.

Ejemplo: Un 'subalfabeto' del castellano módulo 27 consta de 5 caracteres: A, E, O, S, y T, todos ellos equiprobables. Esto en principio puede aceptarse como representativo del lenguaje; es más o menos cierto porque todas esas letras tienen una frecuencia alta en castellano.

La pregunta es: ¿cuántos mensaje de longitud 4 existen y cuántos con sentido?

Solución:

$$R = \log_2 5 = 2{,}3219. \text{ Existirán así } 2^{R*4} = 2^{2{,}3219*4} = 625 = 5^4 \text{ mensajes en total.}$$

Como 1,2 < r < 1,5 entonces cabe esperar x mensajes con sentido de longitud 4 del orden:

$$2^{1{,}2*4} < x < 2^{1{,}5*4} \text{ es decir } 27 < x < 64.$$

Buscando en un diccionario encontrará las 45 palabras que se indican más abajo y que, casualmente, es el valor medio de x, es decir: (27 + 64)/2 = 45.

■■■

Las palabras con sentido son estas 45: aeta, asas, asea, asee, aseo, ases, asta, atea, atas, ates, ateo, atoa, atoe, atoo, osas, oses, osos, oste, otea, otee, oteo, easo, esas, eses, esos, esta, este esto, etas, tasa, tase, taso, teas, tesa, tese, teso, teta, seas, seso, seta, seto, sosa, sota, sote, soto.

Redundancia del lenguaje
La redundancia D del lenguaje será la diferencia entre la ratio absoluta y la ratio real:

$$D = R - r \ (4{,}75 - 1{,}2) \text{ o } (4{,}75 - 1{,}5)$$

$$3{,}25 < D < 3{,}55$$

¿Qué significa esto?

Que el número de bits extras (*bits redundantes*) necesarios para codificar un mensaje suponiendo un alfabeto de 27 caracteres (codificación con 5 bits puesto que $2^5 = 32$ y $2^4 = 16$) será aproximadamente igual a 3,5.

D/R será un factor proporcional, luego:

$$68{,}42\ \% < \text{Redundancia del Lenguaje (D/R)} < 74{,}73\ \%$$

Observe que de este orden es el valor que se obtiene de reducción en el tamaño del archivo cuando se comprime un documento de texto con programas como PKZIP.

El estudio de *Shannon* demuestra que es la estructura del lenguaje la que produce esta redundancia:

1) Existen diferencias en la frecuencia de aparición de cada una de las letras de un texto, entregando una distribución típica.

2) Existe gran cantidad de digramas comunes (en, es, ...), también muchos trigramas (ado, ida, ...), tetragramas (ando, lado, ...), algunos pentagramas (mente, iendo, ...), etc.

3) Existe una estructuración típica de frases y oraciones con sentido en nuestro lenguaje.

Esto dará pistas al criptoanalista para atacar un sistema.

3. La distancia de unicidad

Se entenderá por Distancia de Unicidad al bloque N de texto cifrado o criptograma mínimo necesario para que se pueda intentar con ciertas expectativas de éxito un ataque en búsqueda de la clave usada para cifrar.

Este valor se obtiene cuando la equivocación de la clave $H_C(K)$ se acerca a cero o tiende a anularse.

A medida que se tenga un criptograma más largo, y por tanto más información, se supone que la tarea de ataque del criptoanalista se va facilitando.

Se busca el tamaño N de criptograma que permita esperar que la solución de K sea única. Suponiendo un cifrador aleatorio, se tiene que:

1) Existirán 2^{RN} mensajes posibles de longitud N.

2) Existirán 2^{rN} mensajes de longitud N con sentido.

3) El espacio de mensajes de longitud N se dividirá en:
   a. Espacio de los mensajes con sentido: $M_{CS} = 2^{rN}$.
   b. Espacio de los mensajes sin sentido: $M_{SS} = 2^{RN} - 2^{rN}$.

4) Los 2^{rN} mensajes con sentido serán equiprobables siendo su valor $p(M_{CS}) = 1/2^{rN} = 2^{-rN}$.

5) El resto de mensajes ($2^{RN} - 2^{rN}$) correspondientes a aquellos sin sentido tendrán una probabilidad nula $p(M_{SS}) = 0$, ya que nunca serán generados.

6) Existirán $2^{H(K)}$ claves equiprobables.

7) En donde H(K) es la entropía de la clave.

8) Con $p(K) = 1/2^{H(K)} = 2^{-H(K)}$.

9) Con estas claves se cifrarán todos los mensajes con sentido dando lugar a 2^{RN} textos cifrados posibles de longitud N.

A diferencia de los mensajes, como es lógico los criptogramas obtenidos serán todos equiprobables.

Esquema para mensajes de longitud N

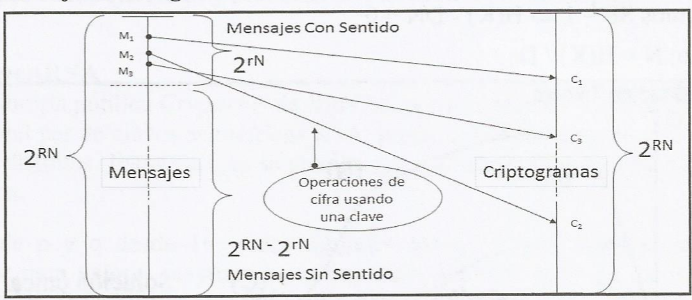


Figura 75: Esquema para mensajes de longitud N.

Escenarios en un cifrador aleatorio de cifra para sólo dos claves k_1 y k_2.

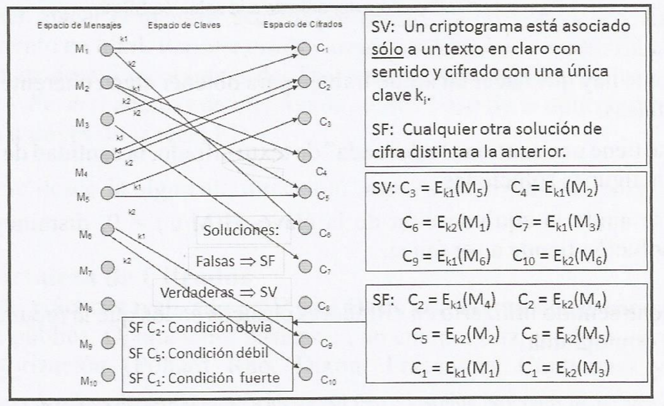


Figura 76: Escenarios en un cifrador aleatorio de cifra para sólo dos claves k_1 y k_2.

Cálculo de la distancia de unicidad

Para cada solución correcta de un texto M cifrado con una clave k del espacio $2^{H(K)}$, existirán otras $(2^{H(K)}-1)$ claves con la misma probabilidad de entregar una solución falta SF.

Sea q la probabilidad de obtener un mensaje con sentido:

$$q = 2^{rN} / 2^{RN} = 2^{(r-R)N} = 2^{-DN}$$

Luego:

$$SF = (2^{H(K)}-1)\, q = (2^{H(K)}-1)\, 2^{-DN} = 2^{H(K)-DN} - 2^{-DN} \quad (\text{como } 2^{-DN} \approx 0)$$

$$SF \approx 2^{H(K)-DN}$$

$$\log_2 SF = H(K) - DN$$

La solución SF = 0 es imposible porque sólo se llega a ella de forma asintótica con un valor de N infinito como se muestra en la imagen siguiente. Se acepta entonces que haya como máximo una sola solución falsa, de ahí su nombre de unicidad, luego:

$$SF = 2^{H(K)-DN}$$

Si hacemos $SF = 1 \Rightarrow H(K) - DN = 0$

Por lo tanto: $N = H(K) / D$

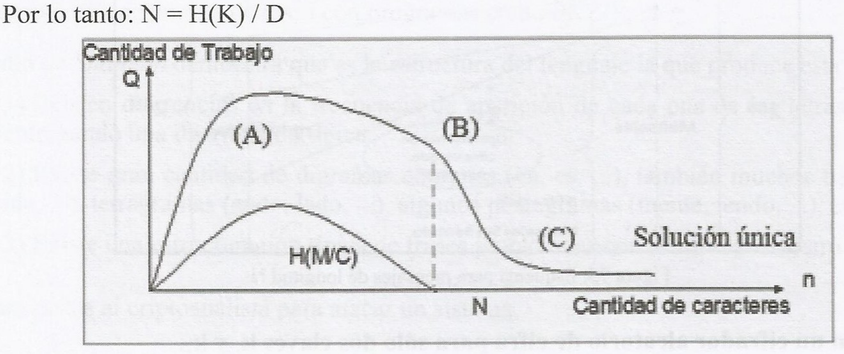


Figura 77: Gráfico correspondiente a la solución SF=0.

a) Inicialmente hay que hacer un arduo trabajo para obtener algo coherente. Aparecen muchas soluciones falsas.

b) Cuando se tiene una cantidad "adecuada" de texto cifrado, la cantidad de trabajo disminuye. Se descartan algunas soluciones.

c) Cuando se anula la equivocación de la clave, H(M/C) = 0, disminuyen las soluciones falsas y la solución tiende a ser única.

Este estudio sólo tiene sentido utilizarlo en cifradores clásicos en los que la redundancia del lenguaje se manifiesta en el criptograma.

Apéndice C
Software educativo

Un buen aprendizaje requiere forzosamente de un proceso guiado mediante el cual se puedan realizar prácticas y ejercicios que refuercen el conocimiento adquirido sobre una materia en concreto. En el caso de la criptografía es necesario utilizar software específico para automatizar el tedioso cálculo matemático implicado en las diferentes operaciones involucradas en los algoritmos a estudiar. Dentro de ese software específico, y siempre desde un punto de vista docente (sino opciones como OpenSSL, GPG, etc., serían las recomendadas) puede destacarse software como *CrypTool*[1].

En el caso de la experimentación con el algoritmo RSA algunos programas recomendados son *genRSA*, *Fortaleza de Cifrados*, *ExpoCrip*, *Dec2Hex*, *RSAManager* y *factor.exe*. Estos programas son los utilizados en los ejercicios y prácticas de este libro. Si lo desea puede experimentar con ellos.

El software genRSA
Basado en la librería pública *Crypto++* de *Wei Dai*, permite realizar los cálculos correspondientes a la generación del par de claves asimétricas RSA, partiendo de dos primos con igual número de bits o ligeramente distintos. Estos valores se pueden generar manualmente o bien de forma automática por el programa.

Para valores de p y q desde 16 -valor mínimo- hasta 32 bits, éstos pueden ser decimales o hexadecimales; para valores mayores hasta una clave de 2048 bits, los datos de la clave deberán ser hexadecimales. Propuesta o calculada automáticamente una clave pública, el programa calcula la clave privada así como todas las claves privadas parejas y el número de mensajes no cifrables. Terminada esta operación, se pueden encontrar los valores de estos mensajes no cifrables, atacar el valor del módulo n por factorización de primos cercanos, atacar la clave privada por la paradoja de cumpleaños o bien atacar el mensaje secreto por medio del cifrado cíclico. En cada caso, la salida genera un documento en html. Permite cifrar y descifrar números o texto. En el caso de números, se incluye la opción de descifrado a través del Teorema del Resto Chino. Incluye un test de primalidad de *Miller-Rabin* y *Fermat* además de una Ayuda sobre el uso de la aplicación y conceptos teóricos, por menú o bien a través de la tecla F1.

Puede descargarse desde la siguiente dirección: *http://www.criptored.upm.es/software/sw_m001d.htm_*

El software Fortaleza de Cifrados
Es un software de prácticas realizado en Visual Basic con herramientas características de sistemas de cifra de clave pública. Operaciones básicas en un cuerpo, raíz, módulo, mcd, inversos, potencia, primalidad, factorización (Pollard Rho, Dixon, Fracciones Continuas) y logaritmo discreto

1 The *CrypTool* project develops the world most-widespread free e-learning programs in the area of cryptography and cryptoanalysis. *http://www.CrypTool.org/en/*

(Búsqueda Exhaustiva, Paso Gigante - Paso Enano, Pohlig - Hellman). Incluye diversos ejemplos y permite el uso de números grandes: centenas de dígitos. Ayuda contextual en formato Windows estándar.

Puede descargarse desde la siguiente dirección: *http://www.criptored.upm.es/software/sw_m001e.htm*

El software ExpoCrip

Es un software para prácticas de cifrado exponencial de números decimales, valores hexadecimales y texto ASCII, realizado con librerías propias y eficiente para operaciones de hasta centenas de bits. Para el sistema RSA, incluye cifra y firma digital, además de la generación manual de claves, entregando en este caso las claves privadas parejas y mensajes no cifrables, así como ataque al módulo n por factorización con método *Pollard Rho*, ataque a la clave privada por la paradoja de cumpleaños y ataque al mensaje secreto por el cifrado cíclico.

En cuanto al sistema *ElGamal* incluye generación de claves, cifra y firma digital, así como su variante de firma DSS, Digital Standard Signature. En este apartado se incluye como ayuda el cálculo de raíz primitiva del cuerpo de forma automática. La aplicación incluye además un menú de herramientas con lista de número primos, lista de número primos fuertes, test de primalidad de *Miller-Rabin*, descomposición de un número por factores primos y cálculo de raíces primitivas. Contempla también una amplia Ayuda contextual del manejo de la aplicación, conceptos teóricos y ejemplos a través de la tecla F1.

Puede descargarse desde la siguiente dirección: *http://www.criptored.upm.es/software/sw_m001l.htm*

El software Dec2Hex

Es un conversor de decimal a hexadecimal y viceversa realizado en Java, especial para la conversión de números grandes de miles de bits.

Para ver las operaciones de conversión del programa y sus créditos, ejecute *Dec2Hex* desde la línea de comandos de la carpeta de Java, por ejemplo si lo ha instalado en c:\criptolab*Dec2Hex*:

C:\Program Files\Java\jre6\bin>java -jar c:\criptolab\Dec2Hex.jar

Puede descargarse desde la siguiente dirección: *http://www.criptored.upm.es/software/sw_m051b.htm*

El software RSA Manager

Programa para sistema operativo Windows de D. *Miguel Schlereth Martínez* que consiste en un Software de laboratorio para la generación automática de claves RSA utilizando el programa OpenSSL. Se pueden generar desde 1 hasta 999 claves, cuyos archivos se guardan en la carpeta de instalación de RSA Manager. Posteriormente la aplicación permite convertir claves en formato texto según comando de OpenSSL para utilizar los parámetros de las mismas en otras aplicaciones. Recuerde que la generación de claves tarda entre 1,5 y 2,0 segundos cada una. La posterior conversión de los archivos a formato texto es prácticamente inmediata.

Puede descargarse desde la siguiente dirección: *http://www.criptored.upm.es/software/sw_m001n.htm*

El software factor.exe

Programa de Shamus Software, Dublin, Irlanda, en formato MS-DOS que permite factorizar números de hasta 150 dígitos. Si desea factorizar números mayores, debe usar la opción -d. Para ayuda, ejecutar el comando factor Enter. Su procedimiento de factorización sigue los siguientes criterios:

1) Primero usa la fuerza bruta para buscar primos pequeños.

2) Después utiliza el método de *Brent*.

3) A continuación sigue con el método de *William*.

4) Luego sigue con el método de *Pollard* (p-1).

5) Y finaliza su intento con el método de la criba cuadrática polinomial múltiple.

6) Si con ello no logra factorizar el número, el programa se rinde.

Puede descargarse desde la siguiente dirección: *http://www.criptored.upm.es/paginas/software.htm#freeware*

Referencias y bibliografía recomendada

Capítulo I. Introducción

1. Auguste Kerckhoffs, 'La cryptographie militaire', Journal des sciences militaires, vol. IX, pp. 5–38, Jan. 1883, pp. 161–191, Feb. 1883. *http://www.petitcolas.net/fabien/kerckhoffs/crypto_militaire_1.pdf*

2. Shannon, Claude E. (July/October 1948). "A Mathematical Theory of Communication". Bell System Technical Journal 27 (3): 379–423. *http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf*

3. Shannon, Claude. "Communication Theory of Secrecy Systems", Bell System Technical Journal, vol. 28(4), page 656–715, 1949. *http://netlab.cs.ucla.edu/wiki/files/shannon1949.pdf*

4. Diffie, W. y M.E.Hellman. "New directions in cryptography", IEEE Transactions on Information Theory 22 (1976), pp. 644-654.

5. C. H. Bennett and G. Brassard, "Quantum Cryptography: Public key distribution and coin tossing", in Proceedings of the IEEE International Conference on Computers, Systems, and Signal Processing, Bangalore, p. 175 (1984)

6. Quantum key distribution. *https://en.wikipedia.org/wiki/Quantum_key_distribution*

Capítulo II. Sistemas de cifra clásica y su evolución a criptosistemas simétricos modernos

1. Edgar Allan Poe. El escarabajo de oro. *http://es.wikipedia.org/wiki/El_escarabajo_de_oro*

2. David Kahn. The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet. Scribner; Rev Sub edition (December 5, 1996). ISBN-10: 0684831309.

3. William F. Friedman. *https://en.wikipedia.org/wiki/Military_Cryptanalytics*

4. Kahn David, Kruh Louis, Mellen Greg, Winkel Brian. Cryptology: Machines, History & Methods. Artech House Cryptology Series. ISBN-13: 978-0890063996.

5. Friedman, William, "The Index of Coincidence and Its Applications in Cryptography", Riverbank Laboratories, publication 22, 1922.

6. Código *Baudot*. *https://es.wikipedia.org/wiki/C%C3%B3digo_Baudot*
Criptoanálisis del cifrado Playfair:

7. Konheim, Alan G., Cryptography: A Primer, John Wiley & Sons, 1981, pp. 95-110.
Libros de referencia para criptografía simétrica y asimétrica (algoritmos de hash, criptosistema de Rabin, transferencia inconsciente, criptoanálisis, etc.)

8. Bruce Schneier. Applied Cryptography (2nd ed). John Wiley & Sons INC, 1995. ISBN 9780471117094.

9. José Pastor Franco, Miguel Ángel Sarasa. Criptografía digital: fundamentos y aplicaciones. Prensas universitarias de Zaragoza, 1998. ISBN: 9788477334910.

10. Raúl Duran Díaz, Luis Hernández Encinas, Jaime Muñoz Masque. El criptosistema RSA, RA-MA, 2005 ISBN 9788478976515.

11. Alfred J. Menezes, Paul C. van Oorschot, Scott A Vanstone. Handbook of Applied Cryptography. CRC Press. ISBN: 0-8493-8523-7. October 1996, 816 pages.

Capítulo III. Criptografía de clave pública: El algoritmo RSA

1. Taher Elgamal. *https://en.wikipedia.org/wiki/ElGamal_encryption*

2. PKCS #1 RSA Cryptography Standard. *https://www.rsa.com/rsalabs/node.asp?id=2125*

3. GCHQ Public Key Cryptography inventors. *http://www.gchq.gov.uk/Press/Pages/100th-IEEE-milestone-award.aspx*

4. Strong Prime. *https://en.wikipedia.org/wiki/Strong_prime*

5. *OpenSSL*: The Open Source toolkit for SSL/TLS. *https://www.openssl.org/*

6. Block cipher modes of operation. *https://en.wikipedia.org/wiki/Block_cipher_modes_of_operation*

7. Esquema de umbral de Shamir e interpolación de Lagrange. Lección 8. Protocolos criptográficos. Protocolo de compartición de secretos. 13:53 minutos. Autor: Dr. Luis Hernández Encinas. CSIC, Madrid. España. *http://www.criptored.upm.es/intypedia/video.php?id=reparto-secretos&lang=es*

8. Hugo D. Scolnik. Fundamentos matemáticos del método RSA. Abril 2004. *http://www-2.dc.uba.ar/materias/crip/docs/rsamath01.pdf*

9. Uwe Hermann. Creating 32768 bit RSA keys for fun and profit. *http://www.hermann-uwe.de/blog/creating-32768-bit-rsa-keys-for-fun-and-profit*

10. Funciones de Bent. *https://en.wikipedia.org/wiki/Bent_function*

11. Andrey Sidorenko, Joachim van den Berg, Remko Foekema, Michiel Grashuis, Jaap de Vos. Bellcore attack in practice. *https://eprint.iacr.org/2012/553.pdf*

12. Durán Díaz, José Raúl (2003) *Números primos especiales y sus aplicaciones criptográficas.* Tesis (Doctoral), E.T.S.I. Telecommunication (UPM). *http://oa.upm.es/193/1/09200317.pdf*

13. P. Lutus. Prime Numbers. Exploring a unique class of numbers. *http://www.arachnoid.com/prime_numbers/index.html*

14. The first 500 primes are listed below, followed by lists of notable types of prime numbers in alphabetical order, giving their respective first terms. *https://en.wikipedia.org/wiki/List_of_prime_numbers*

15. RSA Algorithm, DI Management Services, Australia. Manual online. *http://www.di-mgt.com.au/rsa_alg.html*

Capítulo IV. La seguridad de la criptografía de clave pública y el algoritmo RSA

1. Ley de *Moore. https://en.wikipedia.org/wiki/Moore%27s_law*

2. DES challenges. *https://en.wikipedia.org/wiki/DES_Challenges*

3. DES Cracker machine. *https://w2.eff.org/Privacy/Crypto/Crypto_misc/DESCracker/HTML/19980716_eff_des_faq.html*

4. Brian Randell, The COLOSSUS. (in A History of Computing in the Twentieth Century). *http://www.cs.ncl.ac.uk/research/pubs/books/papers/133.pdf*

5. *Robert D. Silverman*. An Analysis of Shamir's Factoring Device. *https://www.rsa.com/rsalabs/node.asp?id=2089*

6. Luciano Bello, Maximiliano Bertacchini. Predictable RNG in the Vulnerable Debian OpenSSL Package, the What and the How. *http://www.citedef.gob.ar/si6/descargas/openssl-debian-defcon16.pdf*

7. Historical: Cryptographic Challenges: The RSA Factoring Challenge. *https://www.rsa.com/rsalabs/node.asp?id=2093*

8. Dan Boneh. Twenty Years of Attacks on the RSA Cryptosystem (paper)

9. Onur Aciicmez and Cetin Kaya Koc and Jean-Pierre Seifert. On the Power of Simple Branch Prediction Analysis. *https://eprint.iacr.org/2006/351.pdf*

10. Connelly Barnes. Integer Factorization Algorithms. Department of Physics, Oregon State University December 7, 2004. *citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.117.1230&rep=rep1&type=pdf*
RSA-768 has 768 bits (232 decimal digits), and was factored on December 12, 2009 by Thorsten Kleinjung, Kazumaro Aoki, Jens Franke, Arjen K. Lenstra, Emmanuel Thomé, Pierrick Gaudry, Alexander Kruppa, Peter Montgomery, Joppe W. Bos, Dag Arne Osvik, Herman te Riele, Andrey Timofeev, and Paul Zimmermann:

11. Factorization records. *https://en.wikipedia.org/wiki/Integer_factorization_records*

12. Hugo Scolnik. Avances en la Factorización Entera. Ponente invitado en DISI 2007, Día Internacional de la Seguridad de la Información, 3 de diciembre de 2007. E.U. Ingeniería

Técnica de Telecomunicaciones, Universidad Politécnica de Madrid, España. Cátedra UPM Applus+ de Seguridad y Desarrollo Sociedad de la Información. Diapositivas de la charla D. Hugo Scolnik: *http://www.criptored.upm.es/descarga/DISI07_HugoScolnik.zip*. Vídeo realizado por el Gabinete de Tele-Educación GATE de la UPM. *https://www.youtube.com/watch?v=Zrd8tM_1Pnk*

13. Jorge Ramió. Lección 8. Ataque por factorización – Crypt4you. Estudio completo de ataque por factorización - *http://www.criptored.upm.es/crypt4you/temas/RSA/leccion8/leccion08.html*

14. "The world's first commercially available quantum computer". *https://en.wikipedia.org/wiki/D-Wave_Systems.*

15. Verónica Fernández Mármol. "Cómo los ordenadores cuánticos aniquilarían la criptografía actual" VIII Ciclo de Conferencias UPM TASSI. *https://www.youtube.com/watch?v=RYG2XYqJUY0*. Conferencia presentada el 9 de mayo de 2012.

16. Adi Shamir, Eran Tromer. Special-Purpose Hardware for Factoring: the NFS Sieving Step

17. Twinkle. *https://en.wikipedia.org/wiki/TWINKLE*

18. Jorge Ramió Aguirre. Lección 9. Ataque por cifrado cíclico – Crypt4you. Aula virtual de criptografía y seguridad de la información de crypt4you universidad politécnica de Madrid. *http://www.criptored.upm.es/crypt4you/temas/RSA/leccion9/leccion09.html*

19. Paradoja del cumpleaños. *https://en.wikipedia.org/wiki/Birthday_problem.*

20. Riva Richmond. An Attack Sheds Light on Internet Security Holes. *https://www.nytimes.com/2011/04/07/technology/07hack.html?_r=0*

21. ENISA Releases DigiNotar Report: Operation Black Tulip. *http://www.infosecisland.com/blogview/18558-ENISA-Releases-DigiNotar-Report-Operation-Black-Tulip.html*

22. Sergio de los Santos. *http://unaaldia.hispasec.com/2012/06/la-creacion-del-certificado-falso-usado.html.* Hispasec.

23. Brad Arkin. *https://blogs.adobe.com/asset/2012/09/inappropriate-use-of-adobe-code-signing-certificate.html.*

24. CRIME – BEAST attack. *https://en.wikipedia.org/wiki/CRIME_(security_exploit)*

25. Moxie Marlinspike. SSLtrip. 2009 BlackHat DC Presentation on SSL Stripping *http://www.thoughtcrime.org/software/sslstrip/*

26. Moxie Marlinspike. Defcon17: More New Tricks For Defeating SSL In Practice. *https://www.youtube.com/watch?v=ibF36Yyeehw*. Año 2009.

27. Moxie Marlinspike. Notable Research. *https://en.wikipedia.org/wiki/Moxie_Marlinspike*

Índice alfabético

E

F

G

H

I

K

L

M

N

O

P

R

S

T

V

W

Z

Índice de imágenes

Índice de tablas